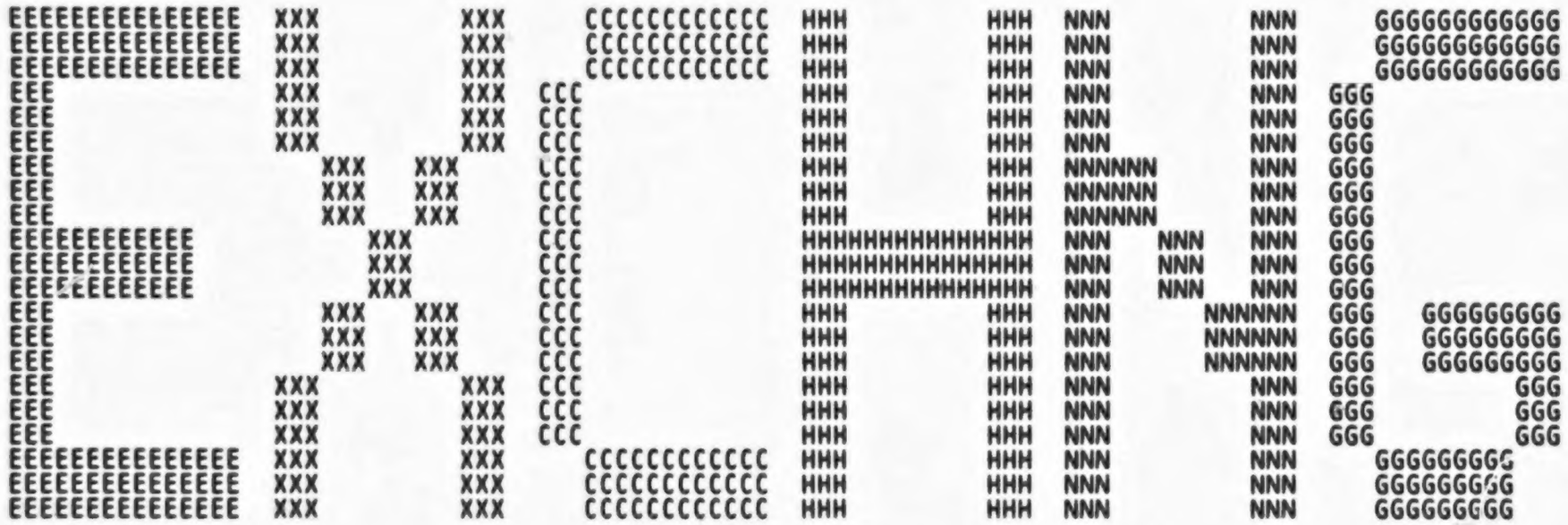
EEEEEEEEEEEEEEE XXX XXX CCCCCCCCCCCCC HHH HHH NNN NNN GGGGGGGGGGGG
EEEEEEEEEEEEEEE XXX XXX CCCCCCCCCCCCC HHH HHH NNN NNN GGGGGGGGGGGG
EEEEEEEEEEEEEEE XXX XXX CCCCCCCCCCCCC HHH HHH NNN NNN GGGGGGGGGGGG
EEE XXX XXX CCC HHH HHH NNN NNN GGG
EEE XXX XXX CCC HHH HHH NNN NNN GGG
EEE XXX XXX CCC HHH HHH NNN NNN GGG
EEE XXX XXX CCC HHH HHH NNNNNN NNN GGG
EEE XXX XXX CCC HHH HHH NNNNNN NNN GGG
EEE XXX XXX CCC HHH HHH NNNNNN NNN GGG
EEEEEEEEEEEE XXX CCC HHHHHHHHHHHHHHH NNN NNN NNN GGG
EEEEEEEEEEEE XXX CCC HHHHHHHHHHHHHHH NNN NNN NNN GGG
EEEEEEEEEEEE XXX CCC HHHHHHHHHHHHHHH NNN NNN NNN GGG
EEE XXX XXX CCC HHH HHH NNN NNNNNN GGG GGGGGGGG
EEE XXX XXX CCC HHH HHH NNN NNNNNN GGG GGGGGGGG
EEE XXX XXX CCC HHH HHH NNN NNNNNN GGG GGGGGGGG
EEE XXX XXX CCC HHH HHH NNN NNN GGG GGG
EEE XXX XXX CCC HHH HHH NNN NNN GGG GGG
EEE XXX XXX CCC HHH HHH NNN NNN GGG GGG
EEEEEEEEEEEEEEE XXX XXX CCCCCCCCCCCC HHH HHH NNN NNN GGGGGGGGG
EEEEEEEEEEEEEEE XXX XXX CCCCCCCCCCCC HHH HHH NNN NNN GGGGGGGGG
EEEEEEEEEEEEEEE XXX XXX CCCCCCCCCCCC HHH HHH NNN NNN GGGGGGGGG

FILEID**EXCCMD

```
EEEEEEEEEE  XX      XX    CCCCCCCC    CCCCCCCC  MM      MM  DDDDDDDD
EEEEEEEEEE  XX      XX    CCCCCCCC    CCCCCCCC  MM      MM  DDDDDDDD
EE          XX      XX  CC          CC          MMMM  MMMM  DD      DD
EE          XX      XX  CC          CC          MMMM  MMMM  DD      DD
EE            XX  XX    CC          CC          MM  MM  MM  DD      DD
EE            XX  XX    CC          CC          MM  MM  MM  DD      DD
EEEEEEEE        XX      CC          CC          MM      MM  DD      DD
EEEEEEEE        XX      CC          CC          MM      MM  DD      DD
EE            XX  XX    CC          CC          MM      MM  DD      DD
EE            XX  XX    CC          CC          MM      MM  DD      DD
EE          XX      XX  CC          CC          MM      MM  DD      DD      ....
EE          XX      XX  CC          CC          MM      MM  DD      DD      ....
EEEEEEEEEE  XX      XX    CCCCCCCC    CCCCCCCC  MM      MM  DDDDDDDD        ....
EEEEEEEEEE  XX      XX    CCCCCCCC    CCCCCCCC  MM      MM  DDDDDDDD        ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

```
;  1     0001  0 MODULE  exch$cmd                                %TITLE 'Command parsing utility routines'
;  2     0002  0                 (
;  3     0003  0                 IDENT = 'V04-000',
;  4     0004  0                 ADDRESSING_MODE (EXTERNAL=LONG_RELATIVE, NONEXTERNAL=WORD_RELATIVE)
;  5     0005  0                 ) =
;  6     0006  1 BEGIN
;  7     0007  1 !
;  8     0008  1 !*****************************************************************************
;  9     0009  1 !*                                                                           *
; 10     0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
; 11     0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
; 12     0012  1 !*  ALL RIGHTS RESERVED.                                                     *
; 13     0013  1 !*                                                                           *
; 14     0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
; 15     0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE  *
; 16     0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER  *
; 17     0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY  *
; 18     0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY  *
; 19     0019  1 !*  TRANSFERRED.                                                             *
; 20     0020  1 !*                                                                           *
; 21     0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE  *
; 22     0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT  *
; 23     0023  1 !*  CORPORATION.                                                             *
; 24     0024  1 !*                                                                           *
; 25     0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS  *
; 26     0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
; 27     0027  1 !*                                                                           *
; 28     0028  1 !*                                                                           *
; 29     0029  1 !*****************************************************************************
; 30     0030  1
; 31     0031  1 !++
; 32     0032  1 ! FACILITY:     EXCHANGE - Foreign volume interchange facility
; 33     0033  1 !
; 34     0034  1 ! ABSTRACT:     Command processing utility routines
; 35     0035  1 !
; 36     0036  1 ! ENVIRONMENT:  VAX/VMS User mode
; 37     0037  1 !
; 38     0038  1 ! AUTHOR:       CW Hobbs                CREATION DATE: 16-July-1982
; 39     0039  1 !
; 40     0040  1 ! MODIFIED BY:
; 41     0041  1 !
; 42     0042  1 !       V03-002 CWH3002         CW Hobbs                    4-Mar-1984
; 43     0043  1 !               Add EXCH$CMD_PARSE_NULL_FILE routine, and call it after $PARSE in
; 44     0044  1 !               EXCH$CMD_PARSE_FILESPEC to make sure that RMS releases all file
; 45     0045  1 !               context.  Also, change EXCH$CMD_RELATED_FILE_PARSE to us NAM$V_SYNCHK
; 46     0046  1 !               syntax check only option so that null file parse will not be needed.
; 47     0047  1 !               During several operations EXCHANGE was leaving extra channels around
; 48     0048  1 !               because channels were being saved by the $PARSE calls.  Fix to give
; 49     0049  1 !               BADPAD error when bogus radix specifier is given.
; 50     0050  1 !
; 51     0051  1 !
; 52     0052  1 !--
; 53     0053  1
; 54     0054  1 ! Include files:
; 55     0055  1 !
; 56     0056  1 MACRO $module_name_string = 'exch$cmd' %;       ! The require file needs to know our module name
; 57     0057  1 REQUIRE 'SRC$:EXCREQ'
```

```
;  58          0058  1     ;
```

```
;    60          0155  1 %SBTTL 'Module table of contents'
;    61          0156  1
;    62          0157  1 ! Module table of contents:
;    63          0158  1 !
;    64          0159  1 FORWARD ROUTINE
;    65          0160  1     exch$cmd_cli_get_integer,                   ! Get integer value from CLI
;    66          0161  1          cmd_convert_uic : NOVALUE,             ! Convert UIC format directory to binary pieces
;    67          0162  1     exch$cmd_fetch_rec_format,                  ! Get /RECORD_FORMAT qualifier
;    68          0163  1     exch$cmd_fetch_recfmt_implied : NOVALUE,    ! Assume record format from file type
;    69          0164  1     exch$cmd_fetch_vol_format,                  ! Get /VOLUME_FORMAT qualifier external entry
;    70          0165  1          cmd_fetch_volfmt_explicit,             ! Get /VOLUME_FORMAT qualifier when explicit
;    71          0166  1          cmd_fetch_volfmt_implied,              ! Assume volume format from device characteristics
;    72          0167  1     exch$cmd_namb_clone,                        ! Make a duplicate of a filled in namb
;    73          0168  1          cmd_namb_fab_copy : NOVALUE,           ! Copy from RMS control block to NAMB block
;    74          0169  1     exch$cmd_match_filename,                    ! Compare actual and target filenames
;    75          0170  1     exch$cmd_parse_filespec,                    ! Fill out a name block for a file parameter
;    76          0171  1     exch$cmd_parse_null_file : NOVALUE,         ! Parse the null string to release RMS channel
;    77          0172  1     exch$cmd_related_file_fixup : NOVALUE,      ! Fixup the name block for an RT-11 magtape file name
;    78          0173  1     exch$cmd_related_file_parse,                ! Fill out a name block for a file parameter
;    79          0174  1     exch$cmd_unwind_cli_syntax                  ! Look for MSG$_SYNTAX signal
;    80          0175  1     ;
;    81          0176  1
;    82          0177  1 ! EXCHANGE facility routines
;    83          0178  1 !
;    84          0179  1 EXTERNAL ROUTINE
;    85          0180  1     exch$main_exit,                             ! Exit routine for end-of-file
;    86          0181  1     exch$util_file_error,                       ! Signal RMS error
;    87          0182  1     exch$util_find_mounted_volb,                ! Checked for mounted volume.
;    88          0183  1     exch$util_namb_allocate,                    ! Allocate a name block
;    89          0184  1     exch$util_namb_release : NOVALUE,           ! Release a name block
;    90          0185  1     exch$util_up_case : NOVALUE jsb_r1r2r3      ! Convert string to uppercase
;    91          0186  1     ;
;    92          0187  1
;    93          0188  1 ! Equated symbols:
;    94          0189  1 !
;    95          0190  1 !LITERAL
;    96          0191  1 !    ;
;    97          0192  1
;    98          0193  1 ! Bound declarations:
;    99          0194  1 !
;   100          0195  1 BIND
;   101          0196  1     ascid_recfmt_pad    = %ASCID 'RECORD_FORMAT.PAD'
;   102          0197  1     ;
```

```
;  104     0198  1 GLOBAL ROUTINE exch$cmd_cli_get_integer (qual_desc : $ref_bblock, ret_value, negflg) = %SBTTL 'exch$cmd_cli_
;  105     0199  2 BEGIN
;  106     0200  2 !++
;  107     0201  2 !
;  108     0202  2 ! FUNCTIONAL DESCRIPTION:
;  109     0203  2 !
;  110     0204  2 !       This routine fetches an integer value for the qualifier described by the input parameter.  The value
;  111     0205  2 !       converted to an integer and returned as the second parameter.
;  112     0206  2 !
;  113     0207  2 ! INPUTS:
;  114     0208  2 !
;  115     0209  2 !       qual_desc  -  the address of a string descriptor for the qualifier name
;  116     0210  2 !       negflg     -  (optional) negative values OK if true, negatives not ok if false or missing
;  117     0211  2 !
;  118     0212  2 ! IMPLICIT INPUTS:
;  119     0213  2 !
;  120     0214  2 !       none
;  121     0215  2 !
;  122     0216  2 ! OUTPUTS:
;  123     0217  2 !
;  124     0218  2 !       ret_value  -  the address of a longword to receive the converted value.  Value is zero if not presen
;  125     0219  2 !                     if an error occurs
;  126     0220  2 !
;  127     0221  2 ! IMPLICIT OUTPUTS:
;  128     0222  2 !
;  129     0223  2 !       none
;  130     0224  2 !
;  131     0225  2 ! ROUTINE VALUE:
;  132     0226  2 !
;  133     0227  2 !       true if able to convert and return value, -1 if not present, error code if error in conversion or va
;  134     0228  2 !
;  135     0229  2 ! SIDE EFFECTS:
;  136     0230  2 !
;  137     0231  2 !       errors will be signalled
;  138     0232  2 !--
;  139     0233  2
;  140     0234  2 LOCAL
;  141     0235  2     value,
;  142     0236  2     status,
;  143     0237  2     tmp_desc    : $desc_block                   ! String to get the ascii text
;  144     0238  2     ;
;  145     0239  2
;  146     0240  2 BUILTIN
;  147     0241  2     ACTUALCOUNT;                                ! Count of input parameters
;  148     0242  2
;  149     0243  2
;  150     0244  2 ! Start by setting our return value to zero
;  151     0245  2 !
;  152     0246  2 .ret_value = 0;
;  153     0247  2
;  154     0248  2 ! Return -1 if the qualifier is not present
;  155     0249  2 !
;  156     0250  2 IF NOT cli$present (.qual_desc)
;  157     0251  2 THEN
;  158     0252  2     RETURN -1;
;  159     0253  2
;  160     0254  2 ! Get the ascii string for the value
```

```
;  161          0255  2 !
;  162          0256  2 $dyn_str_desc_init (tmp_desc);                  ! Set up the desc to the null dynamic string
;  163          0257  3 IF NOT (status = cli$get_value (.qual_desc, tmp_desc))
;  164          0258  2 THEN
;  165          0259  2     $exch_signal_return ($warning_stat (status));       ! Tell about the error
;  166          0260  2
;  167          0261  2 ! Convert the text string to a 32-bit integer
;  168          0262  2 !
;  169          0263  3 IF NOT (status = ots$cvt_ti_l (tmp_desc, value))
;  170          0264  2 THEN
;  171          0265  2     $exch_signal_return (exch$_badvalue, 1, tmp_desc, .status);
;  172          0266  2
;  173          0267  2 ! We can't have negative values unless negflg is present and true
;  174          0268  2 !
;  175          0269  2 IF .value LSS 0
;  176          0270  2 THEN
;  177          0271  3     BEGIN
;  178          0272  3
;  179          0273  3     IF ACTUALCOUNT() LEQ 2                          ! Negflg wasn't specified
;  180          0274  3     THEN
;  181          0275  3         $exch_signal_return (exch$_badvalue, 1, tmp_desc);
;  182          0276  3
;  183          0277  3     IF NOT .negflg                                  ! Negflg is false
;  184          0278  3     THEN
;  185          0279  3         $exch_signal_return (exch$_badvalue, 1, tmp_desc);
;  186          0280  3
;  187          0281  2     END;
;  188          0282  2
;  189          0283  3 IF NOT (status = str$free1_dx (tmp_desc))       ! Release the dynamic string, should never fail
;  190          0284  2 THEN
;  191          0285  2     $exch_signal_stop (.status);
;  192          0286  2
;  193          0287  2 .ret_value = .value;                            ! Pass the value back to the calling routine.
;  194          0288  2
;  195          0289  2 RETURN true;
;  196          0290  1 END;


                                                        .TITLE  EXCH$CMD Command parsing utility routines
                                                        .IDENT  \V04-000\

                                                        .PSECT  EXCH$CMD_PLIT,NOWRT,2

50  2E  54  41  4D  52  4F  46  5F  44  52  4F  43  45  52  00000 P.AAB:   .ASCII  \RECORD_FORMAT.PAD\<0><0><0>                ;
                                         00  00  00  44  41  0000F
                                             010E0011  00014 P.AAA:   .LONG   17694737                                   ;
                                             00000000'  00018         .ADDRESS P.AAB                                    ;

                                                    ASCID_RECFMT_PAD=   P.AAA
                                                        .EXTRN  EXCH$MAIN_EXIT, EXCH$UTIL_FILE_ERROR
                                                        .EXTRN  EXCH$UTIL_FIND_MOUNTED_VOLB
                                                        .EXTRN  EXCH$UTIL_NAMB_ALLOCATE
                                                        .EXTRN  EXCH$UTIL_NAMB_RELEASE
                                                        .EXTRN  EXCH$UTIL_UP_CASE
                                                        .EXTRN  CLI$PRESENT, EXCH$GQ_DYN_STR_TEMPLATE
                                                        .EXTRN  CLI$GET_VALUE, OTS$CVT_TI_L
                                                        .EXTRN  STR$FREE1_DX, LIB$STOP
```

```
                                                        .PSECT  EXCH$CMD_CODE,NOWRT,2

                                            001C 00000          .ENTRY  EXCH$CMD_CLI_GET_INTEGER, Save R2,R3,R4     ; 0198
                    54 00000000G    00   9E 00002          MOVAB   LIB$SIGNAL, R4
                    5E              0C   C2 00009          SUBL2   #12, SP                                     ;
                             08     BC   D4 0000C          CLRL    @RET_VALUE                                  ; 0246
                             04     AC   DD 0000F          PUSHL   QUAL_DESC                                   ; 0250
          00000000G 00              01   FB 00012          CALLS   #1, CLI$PRESENT                             ;
                    04              50   E8 00019          BLBS    R0, 1$                                      ;
                    50              01   CE 0001C          MNEGL   #1, R0                                      ; 0252
                                    04   0001F             RET                                                 ;
                 04 AE 00000000G    EF   7D 00020 1$:      MOVQ    TMPL, DESC                                  ; 0256
                             04     AE   9F 00028          PUSHAB  TMP_DESC                                    ; 0257
                             04     AC   DD 0002B          PUSHL   QUAL_DESC                                   ;
          00000000G 00              02   FB 0002E          CALLS   #2, CLI$GET_VALUE                           ;
                    53              50   D0 00035          MOVL    R0, STATUS                                  ;
                    0D              53   E8 00038          BLBS    STATUS, 2$                                  ;
                    53              07   8A 0003B          BICB2   #7, STATUS2                                 ; 0259
                    52              53   D0 0003E          MOVL    STATUS2, TEMP                               ;
                                    52   DD 00041          PUSHL   TEMP                                        ;
                    64              01   FB 00043          CALLS   #1, LIB$SIGNAL                              ;
                                    45   11 00046          BRB     5$                                          ;
                                    5E   DD 00048 2$:      PUSHL   SP                                          ; 0263
                             08     AE   9F 0004A          PUSHAB  TMP_DESC                                    ;
          00000000G 00              02   FB 0004D          CALLS   #2, OTS$CVT_TI_L                            ;
                    53              50   D0 00054          MOVL    R0, STATUS                                  ;
                    15              53   E8 00057          BLBS    STATUS, 3$                                  ;
                    52 00F81110     8F   D0 0005A          MOVL    #16257296, TEMP                             ; 0265
                                    53   DD 00061          PUSHL   STATUS                                      ;
                             08     AE   9F 00063          PUSHAB  TMP_DESC                                    ;
                                    01   DD 00066          PUSHL   #1                                          ;
                                    52   DD 00068          PUSHL   TEMP                                        ;
                    64              04   FB 0006A          CALLS   #4, LIB$SIGNAL                              ;
                                    1E   11 0006D          BRB     5$                                          ;
                                    6E   D5 0006F 3$:      TSTL    VALUE                                       ; 0269
                                    1E   18 00071          BGEQ    6$                                          ;
                    02              6C   91 00073          CMPB    (AP), #2                                    ; 0273
                                    04   1B 00076          BLEQU   4$                                          ;
                    15        0C    AC   E8 00078          BLBS    NEGFLG, 6$                                  ; 0277
                    52 00F81110     8F   D0 0007C 4$:      MOVL    #16257296, TEMP                             ; 0279
                             04     AE   9F 00083          PUSHAB  TMP_DESC                                    ;
                                    01   DD 00086          PUSHL   #1                                          ;
                                    52   DD 00088          PUSHL   TEMP                                        ;
                    64              03   FB 0008A          CALLS   #3, LIB$SIGNAL                              ;
                    50              52   D0 0008D 5$:      MOVL    TEMP, R0                                    ;
                                    04   00090             RET                                                 ;
                             04     AE   9F 00091 6$:      PUSHAB  TMP_DESC                                    ; 0283
          00000000G 00              01   FB 00094          CALLS   #1, STR$FREE1_DX                            ;
                    53              50   D0 0009B          MOVL    R0, STATUS                                  ;
                    0A              53   E8 0009E          BLBS    STATUS, 7$                                  ;
                                    53   DD 000A1          PUSHL   STATUS                                      ; 0285
          00000000G 00              01   FB 000A3          CALLS   #1, LIB$STOP                                ;
                                    04   000AA             RET                                                 ;
                 08 BC              6E   D0 000AB 7$:      MOVL    VALUE, @RET_VALUE                           ; 0287
                    50              01   D0 000AF          MOVL    #1, R0                                      ; 0289
                                    04   000B2             RET                                                 ; 0290
```

```
; Routine Size:  179 bytes,    Routine Base:  EXCH$CMD_CODE + 0000
```

```
;  198     0291  1 GLOBAL ROUTINE cmd_convert_uic (fabb : $ref_bblock, namb : $ref_bblock) : NOVALUE =     %SBTTL 'cmd_convert_
;  199     0292  2 BEGIN
;  200     0293  2 !++
;  201     0294  2 !
;  202     0295  2 ! FUNCTIONAL DESCRIPTION:
;  203     0296  2 !
;  204     0297  2 !       This routine converts the ASCII uic format directory to binary values, which are then stored in the
;  205     0298  2 !       namb.
;  206     0299  2 !
;  207     0300  2 ! INPUTS:
;  208     0301  2 !
;  209     0302  2 !       fabb - the address of an RMS FAB, assumed to have a nam block
;  210     0303  2 !       namb - address of the created name block
;  211     0304  2 !
;  212     0305  2 ! IMPLICIT INPUTS:
;  213     0306  2 !
;  214     0307  2 !       none
;  215     0308  2 !
;  216     0309  2 ! OUTPUTS:
;  217     0310  2 !
;  218     0311  2 !       none
;  219     0312  2 !
;  220     0313  2 ! IMPLICIT OUTPUTS:
;  221     0314  2 !
;  222     0315  2 !       none
;  223     0316  2 !
;  224     0317  2 ! ROUTINE VALUE:
;  225     0318  2 !
;  226     0319  2 !       True if success, warning status code if unable to parse the parameter
;  227     0320  2 !
;  228     0321  2 ! SIDE EFFECTS:
;  229     0322  2 !
;  230     0323  2 !       $NAMB block is filled with file name information
;  231     0324  2 !--
;  232     0325  2
;  233     0326  2 $dbgtrc_prefix ('cmd_convert_uic> ');
;  234     0327  2
;  235     0328  2 LOCAL
;  236     0329  2     comma,
;  237     0330  2     len,
;  238     0331  2     buf : REF VECTOR [ ,BYTE],
;  239     0332  2     glen,
;  240     0333  2     gbuf : REF VECTOR [ ,BYTE],
;  241     0334  2     mlen,
;  242     0335  2     mbuf : REF VECTOR [ ,BYTE],
;  243     0336  2     value,
;  244     0337  2     tmp_desc    : $desc_block                   ! String to get the ascii text
;  245     0338  2     ;
;  246     0339  2
;  247     0340  2 BIND
;  248     0341  2     nam = .fabb [fab$l_nam]     : $bblock       ! Address of the name block
;  249     0342  2     ;
;  250     0343  2
;  251     0344  2 $debug_print_lit ('entry');
;  252     0345  2
;  253     0346  2 ! Copy the directory string length and address into locals
;  254     0347  2 !
```

```
;  255     0348  2 len = .nam [nam$b_dir];
;  256     0349  2 buf = .nam [nam$l_dir];
;  257     0350  2
;  258     0351  2 ! We assume that the directory is of the format <grp,mem> or [grp,mem], test these assumptions
;  259     0352  2 !
;  260     0353  2 $logic_check (2, (.len GEQU 5), 275);
;  261     0354  2 $logic_check (2, ((.buf [0] EQL %C '<') OR (.buf [0] EQL %C '[')), 276);
;  262     0355  2 $logic_check (2, ((.buf [.len-1] EQL %C '>') OR (.buf [.len-1] EQL %C ']')), 277);
;  263     0356  2
;  264     0357  2 ! Find the comma
;  265     0358  2 !
;  266     0359  2 comma = CH$FIND_CH (.len, .buf, %C ',');
;  267     0360  2 $logic_check (2, (.comma NEQ 0), 278);
;  268     0361  2
;  269     0362  2 ! Derive the address and length of the group and member strings
;  270     0363  2 !
;  271     0364  2 gbuf = .buf + 1;                                ! Skip past the open bracket
;  272     0365  2 glen = .comma - .gbuf;                          ! Length of group is easy
;  273     0366  2 mbuf = .comma + 1;                              ! Member starts one past the comma
;  274     0367  2 mlen = (.len + .buf - 1) - .mbuf;               ! Member length is a little harder.
;  275     0368  2
;  276     0369  2 ! RMS doesn't set the directory bits if the device is non-directory, so we must do it too.
;  277     0370  2 !
;  278     0371  2 IF ((.mlen EQL 1) AND (.mbuf [0] EQL %C '*')) THEN namb [namb$v_wild_member] = true;
;  279     0372  2 IF ((.glen EQL 1) AND (.gbuf [0] EQL %C '*')) THEN namb [namb$v_wild_group]  = true;
;  280     0373  2
;  281     0374  2 ! Convert the text strings to a 32-bit integers, then store as bytes
;  282     0375  2 !
;  283     0376  2 IF NOT .namb [namb$v_wild_member]
;  284     0377  2 THEN
;  285     0378  3     BEGIN
;  286     0379  3     LOCAL
;  287     0380  3         status;
;  288     0381  3     $stat_str_desc_init (tmp_desc, .mlen, .mbuf);
;  289     0382  4     IF NOT (status = ots$cvt_to_l (tmp_desc, value))
;  290     0383  4     THEN
;  291     0384  4         $exch_signal_stop (.status);
;  292     0385  3     namb [namb$b_uic_member] = .value;
;  293     0386  3     END;
;  294     0387  2 IF NOT .namb [namb$v_wild_group]
;  295     0388  2 THEN
;  296     0389  3     BEGIN
;  297     0390  3     LOCAL
;  298     0391  3         status;
;  299     0392  3     $stat_str_desc_init (tmp_desc, .glen, .gbuf);
;  300     0393  4     IF NOT (status = ots$cvt_to_l (tmp_desc, value))
;  301     0394  4     THEN
;  302     0395  4         $exch_signal_stop (.status);
;  303     0396  3     namb [namb$b_uic_group] = .value;
;  304     0397  3     END;
;  305     0398  2
;  306     0399  2 $debug_print_fao ('Input "!AF":  group "!AF", member "!AF"', .len, .buf, .glen, .gbuf, .mlen, .mbuf);
;  307   P 0400  2 $debug_print_fao ('octal grp !OB mem !OB, wild_grp !UL  wild_mem !UL',
;  308   P 0401  2                             .namb [namb$b_uic_group], .namb [namb$b_uic_member],
;  309     0402  2                             .namb [namb$v_wild_group], .namb [namb$v_wild_member]);
;  310     0403  2
;  311     0404  2 RETURN;
```

```
;  312          0405  1 END;


                                                        .EXTRN  EXCH$_BADLOGIC, OTS$CVT_TO_L

                                            03FC 00000          .ENTRY  CMD_CONVERT_UIC, Save R2,R3,R4,R5,R6,R7,R8,-; 0291
                                                                R9
                                59 00000000G  00   9E 00002          MOVAB   OTS$CVT_TO_L, R9
                                58 00000000G  8F   D0 00009          MOVL    #EXCH$_BADLOGIC, R8
                                57 00000000G  00   9E 00010          MOVAB   LIB$STOP, R7
                                5E            0C   C2 00017          SUBL2   #12, SP
                                50        04  AC   D0 0001A          MOVL    FABB, R0                                0341
                                50        28  A0   D0 0001E          MOVL    40(R0), R0
                                54        3A  A0   9A 00022          MOVZBL  58(R0), LEN                             0348
                                52        48  A0   D0 00026          MOVL    72(R0), BUF                             0349
                                05                54   D1 0002A          CMPL    LEN, #5                                 0353
                                                  0C   1E 0002D          BGEQU   1$
                                7E      0113      8F   3C 0002F          MOVZWL  #275, -(SP)
                                                  01   DD 00034          PUSHL   #1
                                                  58   DD 00036          PUSHL   R8
                                67                03   FB 00038          CALLS   #3, LIB$STOP
                                3C                62   91 0003B  1$:     CMPB    (BUF), #60                              0354
                                                  12   13 0003E          BEQL    2$
                             5B 8F                62   91 00040          CMPB    (BUF), #91
                                                  0C   13 00044          BEQL    2$
                                7E      0114      8F   3C 00046          MOVZWL  #276, -(SP)
                                                  01   DD 0004B          PUSHL   #1
                                                  58   DD 0004D          PUSHL   R8
                                67                03   FB 0004F          CALLS   #3, LIB$STOP
                                3E        FF  A442 91   00052  2$:     CMPB    -1(LEN)[BUF], #62                       0355
                                                  14   13 00057          BEQL    3$
                             5D 8F        FF  A442 91   00059          CMPB    -1(LEN)[BUF], #93
                                                  0C   13 0005F          BEQL    3$
                                7E      0115      8F   3C 00061          MOVZWL  #277, -(SP)
                                                  01   DD 00066          PUSHL   #1
                                                  58   DD 00068          PUSHL   R8
                                67                03   FB 0006A          CALLS   #3, LIB$STOP
                             62                   2C   3A 0006D  3$:     LOCC    #44, LEN, (BUF)                         0359
                                                  02   12 00071          BNEQ    4$
                                                  51   D4 00073          CLRL    R1
                                55                51   D0 00075  4$:     MOVL    R1, COMMA
                                                  0C   12 00078          BNEQ    5$                                      0360
                                7E      0116      8F   3C 0007A          MOVZWL  #278, -(SP)
                                                  01   DD 0007F          PUSHL   #1
                                                  58   DD 00081          PUSHL   R8
                                67                03   FB 00083          CALLS   #3, LIB$STOP
                                53        01  A2   9E 00086  5$:     MOVAB   1(R2), GBUF                             0364
                             56 55                53   C3 0008A          SUBL3   GBUF, COMMA, GLEN                       0365
                                51        01  A5   9E 0008E          MOVAB   1(R5), MBUF                             0366
                             50 54                52   C1 00092          ADDL3   BUF, LEN, R0                            0367
                                50                51   C2 00096          SUBL2   MBUF, R0
                                                  50   D7 00099          DECL    MLEN
                                01                50   D1 0009B          CMPL    MLEN, #1                                0371
                                                  0D   12 0009E          BNEQ    6$
                                2A                61   91 000A0          CMPB    (MBUF), #42
                                                  08   12 000A3          BNEQ    6$
```

```
                          52      08     AC  D0 000A5          MOVL    NAMB, R2                                        ;
                      6C  A2             20  88 000A9          BISB2   #32, 108(R2)                                    ;
                          01             56  D1 000AD 6$:      CMPL    GLEN, #1                                        ; 0372
                                         0D  12 000B0          BNEQ    7$                                              ;
                          2A             63  91 000B2          CMPB    (GBUF), #42                                     ;
                                         08  12 000B5          BNEQ    7$                                              ;
                          52      08     AC  D0 000B7          MOVL    NAMB, R2                                        ;
                      6C  A2             10  88 000BB          BISB2   #16, 108(R2)                                    ;
                          52      08     AC  D0 000BF 7$:      MOVL    NAMB, R2                                        ; 0376
                1E    6C  A2             05  E0 000C3          BBS     #5, 108(R2), 8$                                 ;
                      06  AE      010E    8F  B0 000C8          MOVW    #270, DESC+2                                    ; 0381
                      04  AE             50  B0 000CE          MOVW    MLEN, DESC                                      ;
                      08  AE             51  D0 000D2          MOVL    MBUF, DESC+4                                    ;
                                         5E  DD 000D6          PUSHL   SP                                              ; 0382
                              08     AE  9F  D0 000D8          PUSHAB  TMP_DESC                                        ;
                          69             02  FB 000DB          CALLS   #2, OTS$CVT_TO_L                                ;
                          23             50  E9 000DE          BLBC    STATUS, 9$                                      ;
                    0083  C2             6E  90 000E1          MOVB    VALUE, 131(R2)                                  ; 0385
                24    6C  A2             04  E0 000E6 8$:      BBS     #4, 108(R2), 11$                                ; 0387
                      06  AE      010E    8F  B0 000EB          MOVW    #270, DESC+2                                    ; 0392
                      04  AE             56  B0 000F1          MOVW    GLEN, DESC                                      ;
                      08  AE             53  D0 000F5          MOVL    GBUF, DESC+4                                    ;
                                         5E  DD 000F9          PUSHL   SP                                              ; 0393
                              08     AE  9F  D0 000FB          PUSHAB  TMP_DESC                                        ;
                          69             02  FB 000FE          CALLS   #2, OTS$CVT_TO_L                                ;
                          06             50  E8 00101          BLBS    STATUS, 10$                                     ;
                                         50  DD 00104 9$:      PUSHL   STATUS                                          ; 0395
                          67             01  FB 00106          CALLS   #1, LIB$STOP                                    ;
                                             04 00109          RET                                                     ;
                    0084  C2             6E  90 0010A 10$:     MOVB    VALUE, 132(R2)                                  ; 0396
                                             04 0010F 11$:     RET                                                     ; 0405

; Routine Size:  272 bytes,    Routine Base:  EXCH$CMD_CODE + 00B3
```

```
;  314          0406  1 GLOBAL ROUTINE exch$cmd_fetch_rec_format (namb : $ref_bblock) = %SBTTL 'exch$cmd_fetch_rec_format (namb)'
;  315          0407  2 BEGIN
;  316          0408  2 !++
;  317          0409  2 !
;  318          0410  2 ! FUNCTIONAL DESCRIPTION:
;  319          0411  2 !
;  320          0412  2 !       This routine retrieves the /RECORD_FORMAT qualifier for the current file (i.e. the last CLI$GET_VALU
;  321          0413  2 !       for a filename establishes current).
;  322          0414  2 !
;  323          0415  2 ! INPUTS:
;  324          0416  2 !
;  325          0417  2 !       none
;  326          0418  2 !
;  327          0419  2 ! IMPLICIT INPUTS:
;  328          0420  2 !
;  329          0421  2 !       command language interpreter callbacks will retrieve command line information
;  330          0422  2 !
;  331          0423  2 ! OUTPUTS:
;  332          0424  2 !
;  333          0425  2 !       namb      - record format status will be inserted into the namb
;  334          0426  2 !
;  335          0427  2 ! IMPLICIT OUTPUTS:
;  336          0428  2 !
;  337          0429  2 !       none
;  338          0430  2 !
;  339          0431  2 ! ROUTINE VALUE:
;  340          0432  2 !
;  341          0433  2 !       True if success, bad status code if any error (not present is success)
;  342          0434  2 !
;  343          0435  2 ! SIDE EFFECTS:
;  344          0436  2 !
;  345          0437  2 !       none
;  346          0438  2 !--
;  347          0439  2
;  348          0440  2 $dbgtrc_prefix ('cmd_fetch_rec_format> ');
;  349          0441  2
;  350          0442  2 REGISTER
;  351          0443  2     rec_format,
;  352          0444  2     status
;  353          0445  2     ;
;  354          0446  2
;  355          0447  2 ! We should get a MSG$_SYNTAX error if /RECORD_FORMAT is not allowed on the current parameter.  Simulate a r
;  356          0448  2 ! by unwinding if we see it.
;  357          0449  2 !
;  358          0450  2 ENABLE
;  359          0451  2     exch$cmd_unwind_cli_syntax;
;  360          0452  2
;  361          0453  2 $block_check (2, .namb, namb, 400);              ! It would be nice to know now if the input is something els
;  362          0454  2
;  363          0455  2 namb [namb$b_rec_format] = filb$k_rfmt_invalid; ! Assume that there is no /REC
;  364          0456  2 namb [namb$b_car_control] = filb$k_cctl_cr;      ! Also assume normal carriage-return carriage control
;  365          0457  2
;  366          0458  2 ! Nothing to do if the qualifier wasn't specified
;  367          0459  2 !
;  368          0460  2 IF cli$present (%ASCID 'RECORD_FORMAT')
;  369          0461  2 THEN
;  370          0462  3     BEGIN
```

```
;  371     0463  3
;  372     0464  3      namb [namb$v_rfmt_explicit] = true;          ! Remember that we were given a value explicitly
;  373     0465  3      rec_format = filb$k_rfmt_invalid;            ! Init to the impossible value (0)
;  374     0466  3
;  375     0467  3      ! Convert the keyword presence to the symbolic representation of the qualifier keyword for real record f
;  376     0468  3      !
;  377     0469  3      IF cli$present (%ASCID 'RECORD_FORMAT.BINARY')
;  378     0470  3      THEN
;  379     0471  3          rec_format = filb$k_rfmt_binary;
;  380     0472  3
;  381     0473  3      IF cli$present (%ASCID 'RECORD_FORMAT.FIXED')
;  382     0474  3      THEN
;  383     0475  4          BEGIN
;  384     0476  4
;  385     0477  4          rec_format = filb$k_rfmt_fixed;          ! Save the format code
;  386     0478  4
;  387     0479  4          ! Status will be -1 (suc) if not present, error if present but bad value
;  388     0480  4          !
;  389     0481  5          IF NOT (status = exch$cmd_cli_get_integer (%ASCID 'RECORD_FORMAT.FIXED', namb [namb$l_fixed_len]))
;  390     0482  4          THEN
;  391     0483  4              RETURN .status;
;  392     0484  4
;  393     0485  4          ! If the record is not specified, default the record length to 512
;  394     0486  4          !
;  395     0487  4          IF .namb [namb$l_fixed_len] EQL 0
;  396     0488  4          THEN
;  397     0489  4              namb [namb$l_fixed_len] = 512;
;  398     0490  4
;  399     0491  4          ! If the record is too long, scream and shout
;  400     0492  4          !
;  401     0493  4          IF .namb [namb$l_fixed_len] GTRU filb$s_record_buffer
;  402     0494  4          THEN
;  403     0495  4              $exch_signal_return (exch$_notvallen, 1, .namb [namb$l_fixed_len]);
;  404     0496  4
;  405     0497  3          END;
;  406     0498  3
;  407     0499  3
;  408     0500  3      IF cli$present (%ASCID 'RECORD_FORMAT.STREAM')
;  409     0501  3      THEN
;  410     0502  3          rec_format = filb$k_rfmt_stream;                 ! Set the primary format
;  411     0503  3
;  412     0504  3      ! Get the pad character
;  413     0505  3      !
;  414     0506  3      IF cli$present (ascid_recfmt_pad)
;  415     0507  3      THEN
;  416     0508  4          BEGIN
;  417     0509  4          LOCAL
;  418     0510  4              tmp_desc : $desc_block;
;  419     0511  4          $dyn_str_desc_init (tmp_desc);
;  420     0512  5          IF NOT (status = cli$get_value (ascid_recfmt_pad, tmp_desc))
;  421     0513  4          THEN
;  422     0514  4              RETURN .status;
;  423     0515  4          IF .tmp_desc [dsc$w_length] EQL 1
;  424     0516  4          THEN
;  425     0517  4              namb [namb$b_pad_char] = CH$RCHAR (.tmp_desc [dsc$a_pointer])
;  426     0518  4          ELSE
;  427     0519  5              BEGIN
```

```
;  428     0520  5              LOCAL
;  429     0521  5                  char,
;  430     0522  5                  value;
;  431     0523  5              IF .tmp_desc [dsc$w_length] LSS 3
;  432     0524  5              THEN
;  433     0525  5                  RETURN exch$_badpad;
;  434     0526  5              tmp_desc [dsc$w_length] = .tmp_desc [dsc$w_length] - 2;
;  435     0527  5              char = CH$RCHAR_A (tmp_desc [dsc$a_pointer]);
;  436     0528  5              IF .char NEQ %C'%'
;  437     0529  5              THEN
;  438     0530  5                  RETURN exch$_badpad;
;  439     0531  5              char = CH$RCHAR_A (tmp_desc [dsc$a_pointer]);
;  440     0532  5              SELECTONE .char OF
;  441     0533  5              SET
;  442     0534  5                  [%C'D'] :       status = ots$cvt_ti_l (tmp_desc, value);
;  443     0535  5                  [%C'O'] :       status = ots$cvt_to_l (tmp_desc, value);
;  444     0536  5                  [%C'X'] :       status = ots$cvt_tz_l (tmp_desc, value);
;  445     0537  5                  [OTHERWISE] :   RETURN exch$_badpad;
;  446     0538  5              TES;
;  447     0539  5              IF NOT .status
;  448     0540  5              THEN
;  449     0541  5                  RETURN .status;
;  450     0542  5              namb [namb$b_pad_char] = .value;
;  451     0543  4              END;
;  452     0544  3          END;
;  453     0545  3
;  454     0546  3      ! If we have seen a valid record format, then store the final info in the namb
;  455     0547  3      !
;  456     0548  3      IF .rec_format NEQ filb$k_rfmt_invalid
;  457     0549  3      THEN
;  458     0550  3          namb [namb$b_rec_format] = .rec_format; ! store the final value
;  459     0551  3
;  460     0552  2      END;
;  461     0553  2
;  462     0554  2      ! Get the transfer mode qualifier
;  463     0555  2      !
;  464     0556  2      IF cli$present (%ASCID 'TRANSFER_MODE.AUTOMATIC')       ! Usual case
;  465     0557  2      THEN
;  466     0558  2          namb [namb$b_transfer_mode] = filb$k_xfrm_automatic
;  467     0559  2      ELSE IF cli$present (%ASCID 'TRANSFER_MODE.BLOCK')
;  468     0560  2      THEN
;  469     0561  2          namb [namb$b_transfer_mode] = filb$k_xfrm_block
;  470     0562  2      ELSE IF cli$present (%ASCID 'TRANSFER_MODE.RECORD')
;  471     0563  2      THEN
;  472     0564  2          namb [namb$b_transfer_mode] = filb$k_xfrm_record;
;  473     0565  2
;  474     0566  2      ! Look for the carriage control, again we will unwind if it isn't allowed
;  475     0567  2      !
;  476     0568  4      (BEGIN LOCAL temp;
;  477     0569  4      temp = cli$present (%ASCID 'CARRIAGE_CONTROL');
;  478     0570  4      namb [namb$v_cctl_explicit] = ((.temp EQL cli$_locpres) OR (.temp EQL cli$_locneg));
;  479     0571  4      $debug_print_fao ('temp !XL, cctl_expl !UL', .temp, .namb [namb$v_cctl_explicit]);
;  480     0572  4
;  481     0573  4      ! IF cli$present (%ASCID 'CARRIAGE_CONTROL.CARRIAGE_RETURN')    ! This is the default, as set at entry above
;  482     0574  4      ! THEN
;  483     0575  4      !     namb [namb$b_car_control] = filb$k_cctl_cr;
;  484     0576  4
```

```
;  485          0577  4 IF cli$present (%ASCID 'CARRIAGE_CONTROL.FORTRAN')
;  486          0578  4 THEN
;  487          0579  4     namb [namb$b_car_control] = filb$k_cctl_fortran;
;  488          0580  4
;  489          0581  4 IF cli$present (%ASCID 'CARRIAGE_CONTROL.NONE')
;  490          0582  4  OR
;  491          0583  5    (NOT .temp)
;  492          0584  4 THEN
;  493          0585  4     namb [namb$b_car_control] = filb$k_cctl_none;
;  494          0586  2 END);
;  495          0587  2
;  496          0588  2 RETURN true;
;  497          0589  1 END;


                                                                .PSECT  EXCH$CMD_PLIT,NOWRT,2

00  00  54  41  4D  52  4F  46  5F  44  52  4F  43  45  52  0001C P.AAD:  .ASCII  \RECORD_FORMAT\<0><0><0>                ;
                                                        00  0002B                                                      ;
                                                    010E000D  0002C P.AAC:  .LONG   17694733                           ;
                                                    00000000' 00030         .ADDRESS P.AAD                             ;
42  2E  54  41  4D  52  4F  46  5F  44  52  4F  43  45  52  00034 P.AAF:  .ASCII  \RECORD_FORMAT.BINARY\                ;
                                                59  52  41  4E  49  00043                                              ;
                                                    010E0014  00048 P.AAE:  .LONG   17694740                           ;
                                                    00000000' 0004C         .ADDRESS P.AAF                             ;
46  2E  54  41  4D  52  4F  46  5F  44  52  4F  43  45  52  00050 P.AAH:  .ASCII  \RECORD_FORMAT.FIXED\<0>               ;
                                                00  44  45  58  49  0005F                                              ;
                                                    010E0013  00064 P.AAG:  .LONG   17694739                           ;
                                                    00000000' 00068         .ADDRESS P.AAH                             ;
46  2E  54  41  4D  52  4F  46  5F  44  52  4F  43  45  52  0006C P.AAJ:  .ASCII  \RECORD_FORMAT.FIXED\<0>               ;
                                                00  44  45  58  49  0007B                                              ;
                                                    010E0013  00080 P.AAI:  .LONG   17694739                           ;
                                                    00000000' 00084         .ADDRESS P.AAJ                             ;
53  2E  54  41  4D  52  4F  46  5F  44  52  4F  43  45  52  00088 P.AAL:  .ASCII  \RECORD_FORMAT.STREAM\                ;
                                                4D  41  45  52  54  00097                                              ;
                                                    010E0014  0009C P.AAK:  .LONG   17694740                           ;
                                                    00000000' 000A0         .ADDRESS P.AAL                             ;
41  2E  45  44  4F  4D  5F  52  45  46  53  4E  41  52  54  000A4 P.AAN:  .ASCII  \TRANSFER_MODE.AUTOMATIC\<0>           ;
                        00  43  49  54  41  4D  4F  54  55  000B3                                                      ;
                                                    010E0017  000BC P.AAM:  .LONG   17694743                           ;
                                                    00000000' 000C0         .ADDRESS P.AAN                             ;
42  2E  45  44  4F  4D  5F  52  45  46  53  4E  41  52  54  000C4 P.AAP:  .ASCII  \TRANSFER_MODE.BLOCK\<0>              ;
                                                00  4B  43  4F  4C  000D3                                              ;
                                                    010E0013  000D8 P.AAO:  .LONG   17694739                           ;
                                                    00000000' 000DC         .ADDRESS P.AAP                             ;
52  2E  45  44  4F  4D  5F  52  45  46  53  4E  41  52  54  000E0 P.AAR:  .ASCII  \TRANSFER_MODE.RECORD\                ;
                                                44  52  4F  43  45  000EF                                              ;
                                                    010E0014  000F4 P.AAQ:  .LONG   17694740                           ;
                                                    00000000' 000F8         .ADDRESS P.AAR                             ;
4F  52  54  4E  4F  43  5F  45  47  41  49  52  52  41  43  000FC P.AAT:  .ASCII  \CARRIAGE_CONTROL\                    ;
                                                            4C  0010B                                                  ;
                                                    010E0010  00100 P.AAS:  .LONG   17694736                           ;
                                                    00000000' 00110         .ADDRESS P.AAT                             ;
4F  52  54  4E  4F  43  5F  45  47  41  49  52  52  41  43  00114 P.AAV:  .ASCII  \CARRIAGE_CONTROL.FORTRAN\             ;
                        4E  41  52  54  52  4F  46  2E  4C  00123                                                      ;
                                                    010E0018  0012C P.AAU:  .LONG   17694744                           ;
```

```
                                00000000'  00130                .ADDRESS P.AAV                                                  ;
4F 52 54 4E 4F 43 5F 45 47 41 49 52 52 41 43  00134 P.AAX:      .ASCII   \CARRIAGE_CONTROL.NONE\<0><0><0>                       ;
                  00 00 00 45 4E 4F 4E 2E 4C  00143
                                010E0015  0014C P.AAW:      .LONG    17694741                                               ;
                                00000000'  00150                .ADDRESS P.AAX                                                  ;

                                                                .EXTRN   EXCH$UTIL_BLOCK_CHECK
                                                                .EXTRN   EXCH$_NOTVALLEN
                                                                .EXTRN   EXCH$_BADPAD, OTS$CVT_TZ_L
                                                                .EXTRN   CLI$_LOCPRES, CLI$_LOCNEG

                                                                .PSECT   EXCH$CMD_CODE,NOWRT,2

                                      00FC  00000               .ENTRY   EXCH$CMD_FETCH_REC_FORMAT, Save R2,R3,R4,-             ; 0406
                                                                         R5,R6,R7
                              57     0000'  CF   9E  00002               MOVAB    ASCID_RECFMT_PAD, R7                          ;
                              56 00000000G  00   9E  00007               MOVAB    CLI$PRESENT, R6                               ;
                              5E            0C   C2  0000E               SUBL2    #12, SP                                       ;
                              6D     01C7   CF   DE  00011               MOVAL    26$, (FP)                                     ; 0407
                              54       04   AC   D0  00016               MOVL     NAMB, R4                                      ; 0453
                              52 010A00F7   8F   D0  0001A               MOVL     #17432823, R2                                 ;
                              51     0190   8F   3C  00021               MOVZWL   #400, R1                                      ;
                              50            54   D0  00026               MOVL     R4, R0                                        ;
                                 00000000G  EF   16  00029               JSB      EXCH$UTIL_BLOCK_CHECK                         ;
                                       7B   A4   94  0002F               CLRB     123(R4)                                       ; 0455
                                       7D   A4   94  00032               CLRB     125(R4)                                       ; 0456
                                       18   A7   9F  00035               PUSHAB   P.AAC                                         ; 0460
                              66            01   FB  00038               CALLS    #1, CLI$PRESENT                               ;
                              03            50   E8  0003B               BLBS     R0, 1$                                        ;
                                        011C  31   0003E               BRW      17$                                           ;
                        0085  C4            01   88  00041 1$:         BISB2    #1, 133(R4)                                   ; 0464
                                            52   D4  00046               CLRL     REC_FORMAT                                    ; 0465
                                       34   A7   9F  00048               PUSHAB   P.AAE                                         ; 0469
                              66            01   FB  0004B               CALLS    #1, CLI$PRESENT                               ;
                              03            50   E9  0004E               BLBC     R0, 2$                                        ;
                              52            01   D0  00051               MOVL     #1, REC_FORMAT                                ; 0471
                                       50   A7   9F  00054 2$:         PUSHAB   P.AAG                                         ; 0473
                              66            01   FB  00057               CALLS    #1, CLI$PRESENT                               ;
                              42            50   E9  0005A               BLBC     R0, 4$                                        ;
                              52            02   D0  0005D               MOVL     #2, REC_FORMAT                                ; 0477
                                       7E   A4   9F  00060               PUSHAB   126(R4)                                       ; 0481
                                       6C   A7   9F  00063               PUSHAB   P.AAI                                         ;
                        FDD2  CF            02   FB  00066               CALLS    #2, EXCH$CMD_CLI_GET_INTEGER                  ;
                              53            50   D0  00068               MOVL     R0, STATUS                                    ;
                              5A            53   E9  0006E               BLBC     STATUS, 6$                                    ;
                                       7E   A4   D5  00071               TSTL     126(R4)                                       ; 0487
                                            06   12  00074               BNEQ     3$                                            ;
                           7E   A4     0200   8F   3C  00076               MOVZWL   #512, 126(R4)                                 ; 0489
                     00000200   8F      7E   A4   D1  0007C 3$:         CMPL     126(R4), #512                                 ; 0493
                                            19   1B  00084               BLEQU    4$                                            ;
                              55 00000000G  8F   D0  00086               MOVL     #EXCH$_NOTVALLEN, TEMP                        ; 0495
                                       7E   A4   DD  0008D               PUSHL    126(R4)                                       ;
                                            01   DD  00090               PUSHL    #1                                            ;
                                            55   DD  00092               PUSHL    TEMP                                          ;
                     00000000G  00          03   FB  00094               CALLS    #3, LIB$SIGNAL                                ;
                              50            55   D0  00098               MOVL     TEMP, R0                                      ;
                                            04   00  0009E               RET                                                    ;
```

```
                                 0088  C7  9F  0009F 4$:     PUSHAB  P.AAK                           ; 0500
                              66       01  FB  000A3         CALLS   #1, CLI$PRESENT
                              03       50  E9  000A6         BLBC    R0, 5$
                              52       03  D0  000A9         MOVL    #3, REC_FORMAT                  ; 0502
                                       57  DD  000AC 5$:     PUSHL   R7                              ; 0506
                              66       01  FB  000AE         CALLS   #1, CLI$PRESENT
                              26       50  E9  000B1         BLBC    R0, 7$
                        04    AE 00000000G   EF  7D  000B4         MOVQ    TMPL, DESC              ; 0511
                                       04    AE  9F  000BC         PUSHAB  TMP_DESC                ; 0512
                                       57  DD  000BF         PUSHL   R7
                00000000G  00          02  FB  000C1         CALLS   #2, CLI$GET_VALUE
                              53       50  D0  000C8         MOVL    R0, STATUS
                              7E       53  E9  000CB 6$:     BLBC    STATUS, 14$
                              01       04  AE  B1  000CE         CMPW    TMP_DESC, #1            ; 0515
                                       08  12  000D2         BNEQ    8$
                        0082  C4       08  BE  90  000D4         MOVB    @TMP_DESC+4, 130(R4)    ; 0517
                                       79  11  000DA 7$:     BRB     16$
                              03       04  AE  B1  000DC 8$:     CMPW    TMP_DESC, #3            ; 0523
                                       5F  1F  000E0         BLSSU   12$
                        04    AE                 02  A2  000E2         SUBW2   #2, TMP_DESC            ; 0526
                              50       08  BE  9A  000E6         MOVZBL  @TMP_DESC+4, CHAR       ; 0527
                                       08  AE  D6  000EA         INCL    TMP_DESC+4
                              25                  50  D1  000ED         CMPL    CHAR, #37               ; 0528
                                       4F  12  000F0         BNEQ    12$
                              50       08  BE  9A  000F2         MOVZBL  @TMP_DESC+4, CHAR       ; 0531
                                       08  AE  D6  000F6         INCL    TMP_DESC+4
                00000044   8F                  50  D1  000F9         CMPL    CHAR, #68               ; 0534
                                       0E  12  00100         BNEQ    9$
                                       5E  DD  00102         PUSHL   SP
                                       08  AE  9F  00104         PUSHAB  TMP_DESC
                00000000G  00          02  FB  00107         CALLS   #2, OTS$CVT_TI_L
                                       2C  11  0010E         BRB     11$
                0000004F   8F                  50  D1  00110 9$:     CMPL    CHAR, #79               ; 0535
                                       0E  12  00117         BNEQ    10$
                                       5E  DD  00119         PUSHL   SP
                                       08  AE  9F  0011B         PUSHAB  TMP_DESC
                00000000G  00          02  FB  0011E         CALLS   #2, OTS$CVT_TO_L
                                       15  11  00125         BRB     11$
                00000058   8F                  50  D1  00127 10$:    CMPL    CHAR, #88               ; 0536
                                       11  12  0012E         BNEQ    12$
                                       5E  DD  00130         PUSHL   SP
                                       08  AE  9F  00132         PUSHAB  TMP_DESC
                00000000G  00          02  FB  00135         CALLS   #2, OTS$CVT_TZ_L
                              53                  50  D0  0013C 11$:    MOVL    R0, STATUS
                                       08  11  0013F         BRB     13$
                              50 00000000G   8F  D0  00141 12$:    MOVL    #EXCH$_BADPAD, R0       ; 0537
                                                  04  00148         RET
                              04                  53  E8  00149 13$:    BLBS    STATUS, 15$             ; 0539
                              50                  53  D0  0014C 14$:    MOVL    STATUS, R0              ; 0541
                                                  04  0014F         RET
                        0082  C4                  6E  90  00150 15$:    MOVB    VALUE, 130(R4)          ; 0542
                                                  52  D5  00155 16$:    TSTL    REC_FORMAT              ; 0548
                                                  04  13  00157         BEQL    17$
                           7B    A4                 52  90  00159         MOVB    REC_FORMAT, 123(R4)     ; 0550
                                       00A8       C7  9F  0015D 17$:    PUSHAB  P.AAM                   ; 0556
                              66                  01  FB  00161         CALLS   #1, CLI$PRESENT
                              05                  50  E9  00164         BLBC    R0, 18$
```

```
                                  7C    A4   94 00167           CLRB    124(R4)                                    ; 0558
                                        1E   11 0016A           BRB     20$                                        ;
                                 00C4   C7   9F 0016C  18$:     PUSHAB  P.AAO                                      ; 0559
                      66                01   FB 00170           CALLS   #1, CLI$PRESENT                            ;
                      06                50   E9 00173           BLBC    R0, 19$                                    ;
                7C    A4                01   90 00176           MOVB    #1, 124(R4)                                ; 0561
                                        0E   11 0017A           BRB     20$                                        ;
                                 00E0   C7   9F 0017C  19$:     PUSHAB  P.AAQ                                      ; 0562
                      66                01   FB 00180           CALLS   #1, CLI$PRESENT                            ;
                      04                50   E9 00183           BLBC    R0, 20$                                    ;
                7C    A4                02   90 00186           MOVB    #2, 124(R4)                                ; 0564
                                 00F8   C7   9F 0018A  20$:     PUSHAB  P.AAS                                      ; 0569
                      66                01   FB 0018E           CALLS   #1, CLI$PRESENT                            ;
                      52                50   D0 00191           MOVL    R0, TEMP                                   ;
                                        51   D4 00194           CLRL    R1                                         ; 0570
           00000000G  8F                52   D1 00196           CMPL    TEMP, #CLI$_LOCPRES                        ;
                                        02   12 0019D           BNEQ    21$                                        ;
                                        51   D6 0019F           INCL    R1                                         ;
                                        50   D4 001A1  21$:     CLRL    R0                                         ;
           00000000G  8F                52   D1 001A3           CMPL    TEMP, #CLI$_LOCNEG                         ;
                                        02   12 001AA           BNEQ    22$                                        ;
                                        50   D6 001AC           INCL    R0                                         ;
                53    50                51   89 001AE  22$:     BISB3   R1, R0, R3                                 ;
  0085   C4     01    01                53   F0 001B2           INSV    R3, #1, #1, 133(R4)                        ;
                                 0118   C7   9F 001B9           PUSHAB  P.AAU                                      ; 0577
                      66                01   FB 001BD           CALLS   #1, CLI$PRESENT                            ;
                      04                50   E9 001C0           BLBC    R0, 23$                                    ;
                7D    A4                01   90 001C3           MOVB    #1, 125(R4)                                ; 0579
                                 0138   C7   9F 001C7  23$:     PUSHAB  P.AAW                                      ; 0581
                      66                01   FB 001CB           CALLS   #1, CLI$PRESENT                            ;
                      03                50   E8 001CE           BLBS    R0, 24$                                    ;
                      04                52   E8 001D1           BLBS    TEMP, 25$                                  ; 0583
                7D    A4                02   90 001D4  24$:     MOVB    #2, 125(R4)                                ; 0585
                      50                01   D0 001D8  25$:     MOVL    #1, R0                                     ; 0588
                                             04 001DB           RET                                                ; 0589
                                           0000 001DC  26$:     .WORD   Save nothing                               ; 0407
                                        7E   D4 001DE           CLRL    -(SP)                                      ;
                                        5E   DD 001E0           PUSHL   SP                                         ;
                      7E         04     AC   7D 001E2           MOVQ    4(AP), -(SP)                               ;
               0000V  CF                03   FB 001E6           CALLS   #3, EXCH$CMD_UNWIND_CLI_SYNTAX             ;
                                             04 001EB           RET                                                ;

; Routine Size:  492 bytes,    Routine Base:  EXCH$CMD_CODE + 01C3
```

```
;  499     0590  1 GLOBAL ROUTINE exch$cmd_fetch_recfmt_implied (filb : $ref_bblock,      %SBTTL 'exch$cmd_fetch_recfmt_implie
;  500     0591  1                                                 type : $ref_bvector) : NOVALUE =
;  501     0592  2 BEGIN
;  502     0593  2 !++
;  503     0594  2 !
;  504     0595  2 ! FUNCTIONAL DESCRIPTION:
;  505     0596  2 !
;  506     0597  2 !       This routine sets the correct /RECORD_FORMAT qualifier for the current file based on the record form
;  507     0598  2 !       the namb and the file type string.
;  508     0599  2 !
;  509     0600  2 ! INPUTS:
;  510     0601  2 !
;  511     0602  2 !       filb    - pointer to file structure
;  512     0603  2 !       type    - address of the three letter file type string
;  513     0604  2 !
;  514     0605  2 ! IMPLICIT INPUTS:
;  515     0606  2 !
;  516     0607  2 !       none
;  517     0608  2 !
;  518     0609  2 ! OUTPUTS:
;  519     0610  2 !
;  520     0611  2 !       none
;  521     0612  2 !
;  522     0613  2 ! IMPLICIT OUTPUTS:
;  523     0614  2 !
;  524     0615  2 !       filb$b_rec_format is set, along with other associated bits
;  525     0616  2 !
;  526     0617  2 ! ROUTINE VALUE:
;  527     0618  2 !
;  528     0619  2 !       none
;  529     0620  2 !
;  530     0621  2 ! SIDE EFFECTS:
;  531     0622  2 !
;  532     0623  2 !       none
;  533     0624  2 !--
;  534     0625  2
;  535     0626  2 $dbgtrc_prefix ('cmd_fetch_recfmt_implied> ');
;  536     0627  2
;  537     0628  2 REGISTER
;  538     0629  2     rec_format
;  539     0630  2     ;
;  540     0631  2
;  541     0632  2 BIND
;  542     0633  2     namb = filb [filb$a_assoc_namb] : $ref_bblock,
;  543     0634  2     volb = filb [filb$a_assoc_volb] : $ref_bblock
;  544     0635  2     ;
;  545     0636  2
;  546     0637  2 $debug_print_lit ('entry');
;  547     0638  2 $block_check (2, .filb, filb, 401);
;  548     0639  2 $block_check (2, .namb, namb, 402);
;  549     0640  2 $block_check_if_nonzero (2, .volb, volb, 403);
;  550     0641  2
;  551     0642  2 ! Copy everything from the namb to the filb
;  552     0643  2 !
;  553     0644  2 filb [filb$v_rfmt_explicit]     = .namb [namb$v_rfmt_explicit];
;  554     0645  2 filb [filb$v_cctl_explicit]     = .namb [namb$v_cctl_explicit];
;  555     0646  2
```

```
;  556          0647  2 filb [filb$b_car_control]       = .namb [namb$b_car_control];
;  557          0648  2 filb [filb$b_rec_format]        = .namb [namb$b_rec_format];
;  558          0649  2 filb [filb$b_transfer_mode]     = .namb [namb$b_transfer_mode];
;  559          0650  2 filb [filb$l_fixed_len]         = .namb [namb$l_fixed_len];
;  560          0651  2 filb [filb$b_pad_char]          = .namb [namb$b_pad_char];
;  561          0652  2
;  562          0653  2 ! If the record format is valid, then we are done
;  563          0654  2 !
;  564          0655  2 IF .filb [filb$b_rec_format] NEQ filb$k_rfmt_invalid
;  565          0656  2 THEN
;  566          0657  2     RETURN;
;  567          0658  2
;  568          0659  2 ! Assume a record format based on the volume format and file type string
;  569          0660  2 !
;  570          0661  2 !
;  571          0662  2 CASE .namb [namb$b_vol_format] FROM volb$k_vfmt_lobound TO volb$k_vfmt_hibound OF
;  572          0663  2 SET
;  573          0664  3     [volb$k_vfmt_dos11, volb$k_vfmt_rt11] :     !\ , volb$k_vfmt_rtmt] :
;  574          0665  3         BEGIN
;  575          0666  3         SELECTONE true
;  576          0667  3         OF
;  577          0668  3            SET
;  578          0669  3
;  579          0670  3                [CH$EQL (3, UPLIT BYTE ('OBJ'), 3, type [0])  OR
;  580          0671  3                 CH$EQL (3, UPLIT BYTE ('STB'), 3, type [0])  OR
;  581          0672  3                 CH$EQL (3, UPLIT BYTE ('BIN'), 3, type [0])  OR
;  582          0673  3                 CH$EQL (3, UPLIT BYTE ('LDA'), 3, type [0]) ] :
;  583          0674  3
;  584          0675  3                     filb [filb$b_rec_format] = filb$k_rfmt_binary;
;  585          0676  3
;  586          0677  3                [CH$EQL (3, UPLIT BYTE ('EXE'), 3, type [0])  OR
;  587          0678  3                 CH$EQL (2, UPLIT BYTE ( 'LB'), 2, type [1])  OR
;  588          0679  3                 CH$EQL (3, UPLIT BYTE ('SAV'), 3, type [0])  OR
;  589          0680  3                 CH$EQL (3, UPLIT BYTE ('SML'), 3, type [0])  OR
;  590          0681  3                 CH$EQL (3, UPLIT BYTE ('SYS'), 3, type [0])  OR
;  591          0682  3                 CH$EQL (3, UPLIT BYTE ('TSK'), 3, type [0]) ] :
;  592          0683  3
;  593          0684  4                     BEGIN
;  594          0685  4                     filb [filb$b_rec_format] = filb$k_rfmt_fixed;
;  595          0686  4                     filb [filb$l_fixed_len] = 512;
;  596          0687  3                     END;
;  597          0688  3
;  598          0689  3                [OTHERWISE] :
;  599          0690  3
;  600          0691  3                     filb [filb$b_rec_format]  = filb$k_rfmt_stream;
;  601          0692  3
;  602          0693  3                TES;
;  603          0694  3
;  604          0695  2            END;
;  605          0696  2     [OUTRANGE,INRANGE]          : $logic_check (0, (false), 229);
;  606          0697  2 TES;
;  607          0698  2
;  608          0699  2 RETURN;
;  609          0700  1 END;
```

```
                                                        .PSECT  EXCH$CMD_PLIT,NOWRT,2

                                                4A 42 4F 00154 P.AAY:   .ASCII  \OBJ\                                   ;
                                                42 54 53 00157 P.AAZ:   .ASCII  \STB\                                   ;
                                                4E 49 42 0015A P.ABA:   .ASCII  \BIN\                                   ;
                                                41 44 4C 0015D P.ABB:   .ASCII  \LDA\                                   ;
                                                45 58 45 00160 P.ABC:   .ASCII  \EXE\                                   ;
                                                   42 4C 00163 P.ABD:   .ASCII  \LB\                                    ;
                                                56 41 53 00165 P.ABE:   .ASCII  \SAV\                                   ;
                                                4C 4D 53 00168 P.ABF:   .ASCII  \SML\                                   ;
                                                53 59 53 0016B P.ABG:   .ASCII  \SYS\                                   ;
                                                4B 53 54 0016E P.ABH:   .ASCII  \TSK\                                   ;


                                                        .PSECT  EXCH$CMD_CODE,NOWRT,2

                                                   07FC 00000           .ENTRY  EXCH$CMD_FETCH_RECFMT_IMPLIED, Save R2,R3,- ; 0590
                                                                                R4,R5,R6,R7,R8,R9,R10                       ;
                                             5A 00000000G   EF   9E 00002       MOVAB   EXCH$UTIL_BLOCK_CHECK, R10          ;
                                             59     0000'   CF   9E 00009       MOVAB   P.AAY, R9                           ;
                                             54        04   AC   D0 0000E       MOVL    FILB, R4                            ; 0633
                                             52 035B00FA   8F   D0 00012       MOVL    #56295674, R2                       ; 0638
                                             51      0191  8F   3C 00019       MOVZWL  #401, R1                            ;
                                             50            54   D0 0001E       MOVL    R4, R0                              ;
                                                           6A   16 00021       JSB     EXCH$UTIL_BLOCK_CHECK               ;
                                             53        18  A4   D0 00023       MOVL    24(R4), R3                          ; 0639
                                             52 010A00F7   8F   D0 00027       MOVL    #17432823, R2                       ;
                                             51      0192  8F   3C 0002E       MOVZWL  #402, R1                            ;
                                             50            53   D0 00033       MOVL    R3, R0                              ;
                                                           6A   16 00036       JSB     EXCH$UTIL_BLOCK_CHECK               ;
                                                        1C  A4   D5 00038       TSTL    28(R4)                              ; 0640
                                                           12   13 0003B       BEQL    1$                                  ;
                                                   52 041B00F3   8F   D0 0003D       MOVL    #68878579, R2                       ;
                                                   51      0193  8F   3C 00044       MOVZWL  #403, R1                            ;
                                                   50        1C  A4   D0 00049       MOVL    28(R4), R0                          ;
                                                           6A   16 0004D       JSB     EXCH$UTIL_BLOCK_CHECK               ;
                                          2B   A4        01      00     0085  C3   F0 0004F 1$:    INSV    133(R3), #0, #1, 43(R4)             ; 0644
                                             50      0085  C3      01            01   EF 00057       EXTZV   #1, #1, 133(R3), R0                 ; 0645
                                          2B   A4        01      01            50   F0 0005E       INSV    R0, #1, #1, 43(R4)                  ;
                                                           58        28  A4   9E 00064       MOVAB   40(R4), R8                          ; 0648
                                                           68        7B  A3   90 00068       MOVB    123(R3), (R8)                       ;
                                                      29   A4        7C  A3   B0 0006C       MOVW    124(R3), 41(R4)                     ; 0649
                                                      35   A4        7E  A3   D0 00071       MOVL    126(R3), 53(R4)                     ; 0650
                                                      39   A4      0082  C3   90 00076       MOVB    130(R3), 57(R4)                     ; 0651
                                                                         68   95 0007C       TSTB    (R8)                                ; 0655
                                                                         01   13 0007E       BEQL    2$                                  ;
                                                                            04 00080       RET                                         ;
                                                      03      00      7A   A3   8F 00081 2$:    CASEB   122(R3), #0, #3                     ; 0662
                                             001C            0008            001C            0008        00086 3$:    .WORD   4$-3$,-                     ;
                                                                                                                         5$-3$,-                     ;
                                                                                                                         4$-3$,-                     ;
                                                                                                                         5$-3$                       ;
                                                           7E        E5   8F   9A 0008E 4$:    MOVZBL  #229, -(SP)                         ; 0696
                                                                         01   DD 00092       PUSHL   #1                                  ;
                                                                00000000G   8F   DD 00094       PUSHL   #EXCH$_BADLOGIC                     ;
                                                      00000000G   00            03   FB 0009A       CALLS   #3, LIB$STOP                        ;
```

```
                                                 04 000A1          RET
                                55       08    AC D0 000A2 5$:    MOVL    TYPE, R5                                    ;0670
                                                 57 D4 000A6        CLRL    R7
                           65        69          03 29 000A8        CMPC3   #3, P.AAY, (R5)
                                                 02 12 000AC        BNEQ    6$
                                                 57 D6 000AE        INCL    R7
                                                 56 D4 000B0 6$:    CLRL    R6                                          ;0671
                           65   03   A9          03 29 000B2        CMPC3   #3, P.AAZ, (R5)
                                                 02 12 000B7        BNEQ    7$
                                                 56 D6 000B9        INCL    R6
                                56               57 C8 000BB 7$:    BISL2   R7, R6
                                                 57 D4 000BE        CLRL    R7                                          ;0672
                           65   06   A9          03 29 000C0        CMPC3   #3, P.ABA, (R5)
                                                 02 12 000C5        BNEQ    8$
                                                 57 D6 000C7        INCL    R7
                                57               56 C8 000C9 8$:    BISL2   R6, R7
                                                 56 D4 000CC        CLRL    R6                                          ;0673
                           65   09   A9          03 29 000CE        CMPC3   #3, P.ABB, (R5)
                                                 02 12 000D3        BNEQ    9$
                                                 56 D6 000D5        INCL    R6
                                56               57 C8 000D7 9$:    BISL2   R7, R6
                                01               56 D1 000DA        CMPL    R6, #1                                      ;0672
                                                 04 12 000DD        BNEQ    10$
                                68               01 90 000DF        MOVB    #1, (R8)                                    ;0675
                                                 04 000E2          RET
                                                 57 D4 000E3 10$:   CLRL    R7                                          ;0677
                           65   0C   A9          03 29 000E5        CMPC3   #3, P.ABC, (R5)
                                                 02 12 000EA        BNEQ    11$
                                                 57 D6 000EC        INCL    R7
                                                 56 D4 000EE 11$:   CLRL    R6                                          ;0678
                                01   A5    0F    A9 B1 000F0        CMPW    P.ABD, 1(R5)
                                                 02 12 000F5        BNEQ    12$
                                                 56 D6 000F7        INCL    R6
                                56               57 C8 000F9 12$:   BISL2   R7, R6
                                                 57 D4 000FC        CLRL    R7                                          ;0679
                           65   11   A9          03 29 000FE        CMPC3   #3, P.ABE, (R5)
                                                 02 12 00103        BNEQ    13$
                                                 57 D6 00105        INCL    R7
                                57               56 C8 00107 13$:   BISL2   R6, R7
                                                 56 D4 0010A        CLRL    R6                                          ;0680
                           65   14   A9          03 29 0010C        CMPC3   #3, P.ABF, (R5)
                                                 02 12 00111        BNEQ    14$
                                                 56 D6 00113        INCL    R6
                                56               57 C8 00115 14$:   BISL2   R7, R6
                                                 57 D4 00118        CLRL    R7                                          ;0681
                           65   17   A9          03 29 0011A        CMPC3   #3, P.ABG, (R5)
                                                 02 12 0011F        BNEQ    15$
                                                 57 D6 00121        INCL    R7
                                57               56 C8 00123 15$:   BISL2   R6, R7
                                                 56 D4 00126        CLRL    R6                                          ;0682
                           65   1A   A9          03 29 00128        CMPC3   #3, P.ABH, (R5)
                                                 02 12 0012D        BNEQ    16$
                                                 56 D6 0012F        INCL    R6
                                56               57 C8 00131 16$:   BISL2   R7, R6
                                01               56 D1 00134        CMPL    R6, #1                                      ;0681
                                                 0A 12 00137        BNEQ    17$
                                68               02 90 00139        MOVB    #2, (R8)                                    ;0685
```

```
                     35   A4      0200    BF   3C 0013C          MOVZWL  #512, 53(R4)                    ; 0686
                                           04 00142          RET                                       ; 0666
                          68              03   90 00143 17$:    MOVB    #3, (R8)                          ; 0691
                                           04 00146          RET                                       ; 0700

; Routine Size:  327 bytes,    Routine Base:  EXCH$CMD_CODE + 03AF
```

```
;  611          0701  1 GLOBAL ROUTINE exch$cmd_fetch_vol_format (namb : $ref_bblock) = %SBTTL 'exch$cmd_fetch_vol_format (namb)'
;  612          0702  2 BEGIN
;  613          0703  2 !++
;  614          0704  2 !
;  615          0705  2 ! FUNCTIONAL DESCRIPTION:
;  616          0706  2 !
;  617          0707  2 !       This routine determines the /VOLUME_FORMAT for the current file (i.e. the last CLI$GET_VALUE for a f
;  618          0708  2 !       establishes current).  Control is dispatched to one of two sub-routines depending on whether an expl
;  619          0709  2 !       /VOLUME_FORMAT is present or whether volume format should be assumed from the device characteristics
;  620          0710  2 !
;  621          0711  2 ! INPUTS:
;  622          0712  2 !
;  623          0713  2 !       namb    - pointer to name structure
;  624          0714  2 !
;  625          0715  2 ! IMPLICIT INPUTS:
;  626          0716  2 !
;  627          0717  2 !       command language interpreter callbacks will retrieve command line information
;  628          0718  2 !
;  629          0719  2 ! OUTPUTS:
;  630          0720  2 !
;  631          0721  2 !       namb      - volume format status will be inserted into the namb
;  632          0722  2 !
;  633          0723  2 ! IMPLICIT OUTPUTS:
;  634          0724  2 !
;  635          0725  2 !       none
;  636          0726  2 !
;  637          0727  2 ! ROUTINE VALUE:
;  638          0728  2 !
;  639          0729  2 !       True if success, bad status code if any error (not present is success)
;  640          0730  2 !
;  641          0731  2 ! SIDE EFFECTS:
;  642          0732  2 !
;  643          0733  2 !       none
;  644          0734  2 !--
;  645          0735  2
;  646          0736  2 $dbgtrc_prefix ('cmd_fetch_vol_format> ');
;  647          0737  2
;  648          0738  2 ! We should get a MSG$_SYNTAX error if /VOLUME_FORMAT is not allowed on the current parameter.  Simulate a r
;  649          0739  2 ! by unwinding if we see it.  The return status (from this routine) will be -1 (success) if the unwind occur
;  650          0740  2 !
;  651          0741  2 ENABLE
;  652          0742  2     exch$cmd_unwind_cli_syntax;
;  653          0743  2
;  654          0744  2 $block_check (2, .namb, namb, 404);              ! It would be nice to know now if the input is something els
;  655          0745  2
;  656          0746  2 ! If the qualifier is specified, then return the explicit value
;  657          0747  2 !
;  658          0748  2 IF cli$present (%ASCID 'VOLUME_FORMAT')
;  659          0749  2 THEN
;  660          0750  2     RETURN cmd_fetch_volfmt_explicit (.namb)     ! Decode and check the explicit qualifier
;  661          0751  2 ELSE
;  662          0752  2     RETURN cmd_fetch_volfmt_implied (.namb);     ! Imply volume format from the device
;  663          0753  2
;  664          0754  1 END;
```

```
                                                                        .PSECT  EXCH$CMD_PLIT,NOWRT,2

                                                                00171   .BLKB   3
00  00  54  41  4D  52  4F  46  5F  45  4D  55  4C  4F  56     00174 P.ABJ:  .ASCII  \VOLUME_FORMAT\<0><0><0>                                ;
                                                                00183                                                                 ;
                                                    010E000D    00184 P.ABI:  .LONG   17694733                                        ;
                                                    00000000'   00188         .ADDRESS P.ABJ                                          ;


                                                                        .PSECT  EXCH$CMD_CODE,NOWRT,2

                                                        0004    00000         .ENTRY  EXCH$CMD_FETCH_VOL_FORMAT, Save R2              ; 0701
                                                6D      0037    CF  DE  00002 MOVAL   2$, (FP)                                        ; 0702
                                                52  010A00F7    8F  D0  00007 MOVL    #17432823, R2                                   ; 0744
                                                51      0194    8F  3C  0000E MOVZWL  #404, R1                                        ;
                                                50        04    AC  D0  00013 MOVL    NAMB, R0                                        ;
                                                    00000000G   EF  16  00017 JSB     EXCH$UTIL_BLOCK_CHECK                           ;
                                                        0000'   CF  9F  0001D PUSHAB  P.ABI                                           ; 0748
                                    00000000G   00              01  FB  00021 CALLS   #1, CLI$PRESENT                                 ;
                                                09              50  E9  00028 BLBC    R0, 1$                                          ;
                                                          04    AC  DD  0002B PUSHL   NAMB                                            ; 0750
                                        0000V   CF              01  FB  0002E CALLS   #1, CMD_FETCH_VOLFMT_EXPLICIT                   ;
                                                                04  00033 RET                                                        ; 0752
                                                          04    AC  DD  00034 1$:     PUSHL   NAMB                                    ;
                                        0000V   CF              01  FB  00037 CALLS   #1, CMD_FETCH_VOLFMT_IMPLIED                   ;
                                                                04  0003C RET                                                        ; 0754
                                                        0000    0003D 2$:     .WORD   Save nothing                                    ; 0702
                                                      7E    D4  0003F CLRL    -(SP)                                               ;
                                                      5E    DD  00041 PUSHL   SP                                                  ;
                                                7E        04    AC  7D  00043 MOVQ    4(AP), -(SP)                                    ;
                                        0000V   CF              03  FB  00047 CALLS   #3, EXCH$CMD_UNWIND_CLI_SYNTAX                  ;
                                                                04  0004C RET                                                        ;

; Routine Size:  77 bytes,    Routine Base:  EXCH$CMD_CODE + 04F6
```

```
;  666     0755  1 GLOBAL ROUTINE cmd_fetch_volfmt_explicit (namb : $ref_bblock) = %SBTTL 'cmd_fetch_volfmt_explicit (namb)'
;  667     0756  2 BEGIN
;  668     0757  2 !++
;  669     0758  2 !
;  670     0759  2 ! FUNCTIONAL DESCRIPTION:
;  671     0760  2 !
;  672     0761  2 !       This routine retrieves the /VOLUME_FORMAT qualifier for the current file (i.e. the last CLI$GET_VALU
;  673     0762  2 !       for a filename establishes current).
;  674     0763  2 !
;  675     0764  2 ! INPUTS:
;  676     0765  2 !
;  677     0766  2 !       namb    - pointer to name structure
;  678     0767  2 !
;  679     0768  2 ! IMPLICIT INPUTS:
;  680     0769  2 !
;  681     0770  2 !       command language interpreter callbacks will retrieve command line information
;  682     0771  2 !
;  683     0772  2 ! OUTPUTS:
;  684     0773  2 !
;  685     0774  2 !       namb      - volume format status will be inserted into the namb
;  686     0775  2 !
;  687     0776  2 ! IMPLICIT OUTPUTS:
;  688     0777  2 !
;  689     0778  2 !       none
;  690     0779  2 !
;  691     0780  2 ! ROUTINE VALUE:
;  692     0781  2 !
;  693     0782  2 !       True if success, bad status code if any error
;  694     0783  2 !
;  695     0784  2 ! SIDE EFFECTS:
;  696     0785  2 !
;  697     0786  2 !       none
;  698     0787  2 !--
;  699     0788  2
;  700     0789  2 $dbgtrc_prefix ('cmd_fetch_volfmt_explicit> ');
;  701     0790  2
;  702     0791  2 REGISTER
;  703     0792  2     vol_format
;  704     0793  2     ;
;  705     0794  2
;  706     0795  2 $debug_print_lit ('entry> ');
;  707     0796  2
;  708     0797  2 namb [namb$v_vfmt_explicit] = true;              ! Remember that we were given a value explicitly
;  709     0798  2
;  710     0799  2 ! Convert the keyword presence to the symbolic representation of the qualifier keyword for real volume forma
;  711     0800  2 !
;  712     0801  3 vol_format = (SELECTONE cli$_locpres OF
;  713     0802  3                 SET
;  714     0803  3                   [cli$present (%ASCID 'VOLUME_FORMAT.DOS11')] :      volb$k_vfmt_dos11;
;  715     0804  3                   [cli$present (%ASCID 'VOLUME_FORMAT.FILES11')] :    volb$k_vfmt_files11;
;  716     0805  4                   [cli$present (%ASCID 'VOLUME_FORMAT.RT11')] :       BEGIN
;  717     0806  4                                                                       !\  IF .namb [namb$b_devclass] EQL dc$_d
;  718     0807  4                                                                       !\  THEN
;  719     0808  4                                                                       volb$k_vfmt_rt11
;  720     0809  4                                                                       !\  ELSE IF .namb [namb$b_devclass] EQL
;  721     0810  4                                                                       !\  THEN
;  722     0811  4                                                                       !\          vol_format = volb$k_vfmt_rtm
```

```
;  723          0812  3                                                    END;
;  724          0813  3                    [OTHERWISE] :                   $logic_check (0, (false), 308);
;  725          0814  2                  TES);
;  726          0815  2
;  727          0816  2  ! Now we do a cheat.  We will compare our explicit volume format with the implied value, if they are
;  728          0817  2  ! different the explicit value is wrong!  A more involved test will be necessary when/if EXCHANGE
;  729          0818  2  ! supports more than the current set of volume formats.
;  730          0819  2  !
;  731          0820  2  cmd_fetch_volfmt_implied (.namb);
;  732          0821  2  IF .namb [namb$b_vol_format] NEQ .vol_format
;  733          0822  2  THEN
;  734          0823  2      RETURN exch$_invvolfmt;
;  735          0824  2
;  736          0825  2  RETURN true;
;  737          0826  1  END;


                                                                .PSECT  EXCH$CMD_PLIT,NOWRT,2

44 2E 54 41 4D 52 4F 46 5F 45 4D 55 4C 4F 56  0018C P.ABL:   .ASCII  \VOLUME_FORMAT.DOS11\<0>                          ;
                                        00 31 31 53 4F  0019B                                                          ;
                                           010E0013  001A0 P.ABK:   .LONG   17694739                                   ;
                                           00000000'  001A4         .ADDRESS P.ABL                                     ;
46 2E 54 41 4D 52 4F 46 5F 45 4D 55 4C 4F 56  001A8 P.ABN:   .ASCII  \VOLUME_FORMAT.FILES11\<0><0><0>                  ;
                        00 00 00 31 31 53 45 4C 49  001B7                                                              ;
                                           010E0015  001C0 P.ABM:   .LONG   17694741                                   ;
                                           00000000'  001C4         .ADDRESS P.ABN                                     ;
52 2E 54 41 4D 52 4F 46 5F 45 4D 55 4C 4F 56  001C8 P.ABP:   .ASCII  \VOLUME_FORMAT.RT11\<0><0>                       ;
                                        00 00 31 31 54  001D7                                                          ;
                                           010E0012  001DC P.ABO:   .LONG   17694738                                   ;
                                           00000000'  001E0         .ADDRESS P.ABP                                     ;

                                                                .EXTRN  EXCH$_INVVOLFMT

                                                                .PSECT  EXCH$CMD_CODE,NOWRT,2

                                      001C 00000                .ENTRY  CMD_FETCH_VOLFMT_EXPLICIT, Save R2,R3,R4    ; 0755
                 54 00000000G   00    9E 00002                 MOVAB   CLI$PRESENT, R4                             ;
                 53       04    AC    D0 00009                 MOVL    NAMB, R3                                    ; 0797
          0085   C3             04    88 0000D                 BISB2   #4, 133(R3)                                 ;
                 52 00000000G   8F    D0 00012                 MOVL    #CLI$_LOCPRES, R2                           ; 0801
                       0000'    CF    9F 00019                 PUSHAB  P.ABK                                       ; 0803
                 64             01    FB 0001D                 CALLS   #1, CLI$PRESENT                             ;
                 50             52    D1 00020                 CMPL    R2, R0                                      ;
                                05    12 00023                 BNEQ    1$                                          ;
                 52             01    D0 00025                 MOVL    #1, VOL_FORMAT                              ;
                                39    11 00028                 BRB     4$                                          ;
                       0000'    CF    9F 0002A 1$:             PUSHAB  P.ABM                                       ; 0804
                 64             01    FB 0002E                 CALLS   #1, CLI$PRESENT                             ;
                 50             52    D1 00031                 CMPL    R2, R0                                      ;
                                05    12 00034                 BNEQ    2$                                          ;
                 52             02    D0 00036                 MOVL    #2, VOL_FORMAT                              ;
                                28    11 00039                 BRB     4$                                          ;
                       0000'    CF    9F 0003B 2$:             PUSHAB  P.ABO                                       ; 0805
                 64             01    FB 0003F                 CALLS   #1, CLI$PRESENT                             ;
                 50             52    D1 00042                 CMPL    R2, R0                                      ;
```

```
                                                05  12 00045          BNEQ    3$                                          ;
                                          52    03  D0 00047          MOVL    #3, VOL_FORMAT                              ;
                                                17  11 0004A          BRB     4$                                          ;
                                          7E      0134  8F  3C 0004C 3$:     MOVZWL  #308, -(SP)                         ; 0813
                                                01  DD 00051          PUSHL   #1                                          ;
                                            00000000G   8F  DD 00053          PUSHL   #EXCH$_BADLOGIC                     ;
                                 00000000G  00          03  FB 00059          CALLS   #3, LIB$STOP                        ;
                                          52            50  D0 00060          MOVL    R0, VOL_FORMAT                      ;
                                                        53  DD 00063 4$:     PUSHL   R3                                  ; 0820
                                     0000V  CF          01  FB 00065          CALLS   #1, CMD_FETCH_VOLFMT_IMPLIED        ;
          52        7A    A3              08          00  ED 0006A          CMPZV   #0, #8, 122(R3), VOL_FORMAT         ; 0821
                                                        08  13 00070          BEQL    5$                                  ;
                                          50 00000000G  8F  D0 00072          MOVL    #EXCH$_INVVOLFMT, R0                ; 0823
                                                            04 00079          RET                                         ;
                                          50                01  D0 0007A 5$:     MOVL    #1, R0                          ; 0825
                                                            04 0007D          RET                                         ; 0826

; Routine Size:  126 bytes,    Routine Base:  EXCH$CMD_CODE + 0543
```

```
;  739     0827  1 GLOBAL ROUTINE cmd_fetch_volfmt_implied (namb : $ref_bblock) =  %SBTTL 'cmd_fetch_volfmt_implied (namb)'
;  740     0828  2 BEGIN
;  741     0829  2 !++
;  742     0830  2 !
;  743     0831  2 ! FUNCTIONAL DESCRIPTION:
;  744     0832  2 !
;  745     0833  2 !       This routine establishes the default /VOLUME_FORMAT qualifier for the current file based on the devi
;  746     0834  2 !       type and characteristics.
;  747     0835  2 !
;  748     0836  2 ! INPUTS:
;  749     0837  2 !
;  750     0838  2 !       namb    - pointer to name structure
;  751     0839  2 !
;  752     0840  2 ! IMPLICIT INPUTS:
;  753     0841  2 !
;  754     0842  2 !       none
;  755     0843  2 !
;  756     0844  2 ! OUTPUTS:
;  757     0845  2 !
;  758     0846  2 !       namb      - volume format status will be inserted into the namb
;  759     0847  2 !
;  760     0848  2 ! IMPLICIT OUTPUTS:
;  761     0849  2 !
;  762     0850  2 !       none
;  763     0851  2 !
;  764     0852  2 ! ROUTINE VALUE:
;  765     0853  2 !
;  766     0854  2 !       True if success, bad status code if any error (not present is success)
;  767     0855  2 !
;  768     0856  2 ! SIDE EFFECTS:
;  769     0857  2 !
;  770     0858  2 !       none
;  771     0859  2 !--
;  772     0860  2
;  773     0861  2 $dbgtrc_prefix ('cmd_fetch_volfmt_implied> ');
;  774     0862  2
;  775     0863  2 REGISTER
;  776     0864  2     vol_format
;  777     0865  2     ;
;  778     0866  2
;  779     0867  2 BIND
;  780     0868  2     dev = namb [namb$l_fabdev] : $bblock         ! Device characteristics
;  781     0869  2     ;
;  782     0870  2
;  783     0871  2 $trace_print_lit ('entry');
;  784     0872  2
;  785     0873  2 vol_format = volb$k_vfmt_invalid;                !?? assume invalid volume type
;  786     0874  2
;  787     0875  2 ! Determine the characteristics of a disk
;  788     0876  2 !
;  789     0877  2 IF .namb [namb$b_devclass] EQL dc$_disk
;  790     0878  2 THEN
;  791     0879  3     BEGIN
;  792     0880  3
;  793     0881  3     $logic_check (2, (.dev [dev$v_dir] AND .dev [dev$v_fod] AND .dev [dev$v_shr]), 100);
;  794     0882  3
;  795     0883  3     IF   .dev [dev$v_for]
```

```
;   796          0884  4          OR (NOT (.dev [dev$v_mnt]))
;   797          0885  4      THEN
;   798          0886  4          vol_format = volb$k_vfmt_rt11
;   799          0887  4      ELSE IF .dev [dev$v_mnt]
;   800          0888  4      THEN
;   801          0889  4          vol_format = volb$k_vfmt_files11
;   802          0890  4      ELSE
;   803          0891  4          $logic_check (0, (false), 230); ! shouldn't be any other choices
;   804          0892  4
;   805          0893  4      END
;   806          0894  4
;   807          0895  4  ! Determine the characteristics for a tape
;   808          0896  4  !
;   809          0897  4  ELSE IF .namb [namb$b_devclass] EQL dc$_tape
;   810          0898  4  THEN
;   811          0899  4      BEGIN
;   812          0900  4
;   813          0901  4      $logic_check (2, (.dev [dev$v_sqd] AND .dev [dev$v_fod]), 101);
;   814          0902  4
;   815          0903  4      IF    .dev [dev$v_for]
;   816          0904  4        OR (NOT (.dev [dev$v_mnt]))
;   817          0905  4      THEN
;   818          0906  4          vol_format = volb$k_vfmt_dos11
;   819          0907  4      ELSE IF .dev [dev$v_mnt]
;   820          0908  4      THEN
;   821          0909  4          BEGIN
;   822          0910  4          $logic_check (2, (.dev [dev$v_dir] AND .dev [dev$v_sdi]), 102);
;   823          0911  4          vol_format = volb$k_vfmt_files11;
;   824          0912  4          END
;   825          0913  4      ELSE
;   826          0914  4          $logic_check (0, (false), 231);
;   827          0915  4
;   828          0916  4      END
;   829          0917  4
;   830          0918  4  ! Device is neither disk nor tape.  !?? Assume it is Files-11 (term, lp, etc)
;   831          0919  4  !
;   832          0920  2  ELSE
;   833          0921  2      vol_format = volb$k_vfmt_files11;
;   834          0922  2
;   835          0923  2  namb [namb$b_vol_format] = .vol_format;          ! store the final value
;   836          0924  2
;   837          0925  2  RETURN true;
;   838          0926  1  END;


                                                007C 00000          .ENTRY  CMD_FETCH_VOLFMT_IMPLIED, Save R2,R3,R4,R5,-; 0827
                                                                            R6
                                    56 00000000G   8F   D0 00002    MOVL    #EXCH$_BADLOGIC, R6
                                    55 00000000G   00   9E 00009    MOVAB   LIB$STOP, R5
                                    54        04   AC   D0 00010    MOVL    NAMB, R4                                    ; 0868
                                    53        68   A4   9E 00014    MOVAB   104(R4), R3
                                                   52   D4 00018    CLRL    VOL_FORMAT                                  ; 0873
                                    01        78   A4   91 0001A    CMPB    120(R4), #1                                 ; 0877
                                                   24   12 0001E    BNEQ    4$
```

```
                08 63          03 E1 00020          BBC     #3, (R3), 1$                        ; 0881
                04 63          0E E1 00024          BBC     #14, (R3), 1$                       ;
                   0B    02    A3 E8 00028          BLBS    2(R3), 2$                           ;
                   7E    64    8F 9A 0002C 1$:      MOVZBL  #100, -(SP)                         ;
                               01 DD 00030          PUSHL   #1                                  ;
                               56 DD 00032          PUSHL   R6                                  ;
                   65          03 FB 00034          CALLS   #3, LIB$STOP                        ;
                   04    03    A3 E8 00037 2$:      BLBS    3(R3), 3$                           ; 0883
             3E    63          13 E0 0003B          BBS     #19, (R3), 10$                      ; 0884
                   52          03 D0 0003F 3$:      MOVL    #3, VOL_FORMAT                      ; 0886
                               3C 11 00042          BRB     11$                                 ;
                   02    78    A4 91 00044 4$:      CMPB    120(R4), #2                         ; 0897
                               33 12 00048          BNEQ    10$                                 ;
                04 63          05 E1 0004A          BBC     #5, (R3), 5$                        ; 0901
                0B 63          0E E0 0004E          BBS     #14, (R3), 6$                       ;
                   7E    65    8F 9A 00052 5$:      MOVZBL  #101, -(SP)                         ;
                               01 DD 00056          PUSHL   #1                                  ;
                               56 DD 00058          PUSHL   R6                                  ;
                   65          03 FB 0005A          CALLS   #3, LIB$STOP                        ;
                   04    03    A3 E8 0005D 6$:      BLBS    3(R3), 7$                           ; 0903
             05    63          13 E0 00061          BBS     #19, (R3), 8$                       ; 0904
                   52          01 D0 00065 7$:      MOVL    #1, VOL_FORMAT                      ; 0906
                               16 11 00068          BRB     11$                                 ;
                04 63          03 E1 0006A 8$:      BBC     #3, (R3), 9$                        ; 0910
                0B 63          04 E0 0006E          BBS     #4, (R3), 10$                       ;
                   7E    66    8F 9A 00072 9$:      MOVZBL  #102, -(SP)                         ;
                               01 DD 00076          PUSHL   #1                                  ;
                               56 DD 00078          PUSHL   R6                                  ;
                   65          03 FB 0007A          CALLS   #3, LIB$STOP                        ;
                   52          02 D0 0007D 10$:     MOVL    #2, VOL_FORMAT                      ; 0921
             7A    A4          52 90 00080 11$:     MOVB    VOL_FORMAT, 122(R4)                 ; 0923
                   50          01 D0 00084          MOVL    #1, R0                              ; 0925
                               04 00087             RET                                         ; 0926

; Routine Size:  136 bytes,    Routine Base:  EXCH$CMD_CODE + 05C1
```

```
;  840     0927  1 GLOBAL ROUTINE exch$cmd_match_filename (len_fil, adr_fil, len_trg, adr_trg) =   %SBTTL 'exch$cmd_match_filen
;  841     0928  2 BEGIN
;  842     0929  2 !++
;  843     0930  2 !
;  844     0931  2 ! FUNCTIONAL DESCRIPTION:
;  845     0932  2 !
;  846     0933  2 !       This routine compares an RT-11 or DOS-11 filename (result string) with a target filename.  The targe
;  847     0934  2 !       filename may contain the wildcard characters "*" and "%".
;  848     0935  2 !
;  849     0936  2 ! INPUTS:
;  850     0937  2 !
;  851     0938  2 !       len_fil - Length of filename string
;  852     0939  2 !       adr_fil - Address of filename string
;  853     0940  2 !       len_trg - Length of target filename string
;  854     0941  2 !       adr_trg - Address of target filename string
;  855     0942  2 !
;  856     0943  2 ! IMPLICIT INPUTS:
;  857     0944  2 !
;  858     0945  2 !       none
;  859     0946  2 !
;  860     0947  2 ! OUTPUTS:
;  861     0948  2 !
;  862     0949  2 !       none
;  863     0950  2 !
;  864     0951  2 ! IMPLICIT OUTPUTS:
;  865     0952  2 !
;  866     0953  2 !       none
;  867     0954  2 !
;  868     0955  2 ! ROUTINE VALUE:
;  869     0956  2 !
;  870     0957  2 !       True (1) if the target name includes the filename, False (0) otherwise
;  871     0958  2 !
;  872     0959  2 ! SIDE EFFECTS:
;  873     0960  2 !
;  874     0961  2 !       none
;  875     0962  2 !--
;  876     0963  2
;  877     0964  2 $dbgtrc_prefix ('cmd_match_filename> ');
;  878     0965  2
;  879     0966  2 REGISTER
;  880     0967  2     lenf = 2,                                   ! Remaining length of filename
;  881     0968  2     adrf = 3,                                   ! Current address in filename
;  882     0969  2     lent = 4,                                   ! Remaining length of filename
;  883     0970  2     adrt = 5,                                   ! Current address in filename
;  884     0971  2     chf  = 6 : BYTE,                            ! The character from the filename
;  885     0972  2     cht  = 7 : BYTE                             ! The character from target
;  886     0973  2     ;
;  887     0974  2
;  888     0975  2
;  889     0976  2 ! Move the inputs into registers
;  890     0977  2 !
;  891     0978  2 lenf = .len_fil;
;  892     0979  2 adrf = .adr_fil;
;  893     0980  2 lent = .len_trg;
;  894     0981  2 adrt = .adr_trg;
;  895     0982  2
;  896     0983  2 ! Scan through all the characters in the target
```

```
;   897          0984  2 !
;   898          0985  2 DECR k FROM .lent-1 TO 0
;   899          0986  2 DO
;   900          0987  3     BEGIN
;   901          0988  3     lent = .lent-1;                         ! Decrement target length
;   902          0989  3     cht = CH$RCHAR_A(adrt);                 ! Fetch char and bump pointer
;   903          0990  3
;   904          0991  3     IF .cht EQL '*'                         ! Found a wildcard in target
;   905          0992  3     THEN
;   906          0993  4         BEGIN
;   907          0994  4         IF .lent EQL 0                      ! Wildcard at end of target string
;   908          0995  4         THEN                                !  matches everything, return a
;   909          0996  4             RETURN true;                    !  match
;   910          0997  4         DECR i FROM .lenf-1 TO 0            ! Look through rest of filename
;   911          0998  4         DO
;   912          0999  5             BEGIN
;   913          1000  5             IF exch$cmd_match_filename (.lenf, .adrf, .lent, .adrt)     ! Recursively
;   914          1001  5             THEN                            !  examine rest of filename from this
;   915          1002  5                 RETURN true;                !  point in target, return true if found
;   916          1003  5             lenf = .lenf-1;                 ! Advance one character in the filename
;   917          1004  5             adrf = .adrf+1;                 !  and repeat recursive call
;   918          1005  4             END;
;   919          1006  4         RETURN false;                       ! We did not match from wildcard
;   920          1007  4         END
;   921          1008  3     ELSE                                    ! No wildcard in target
;   922          1009  4         BEGIN
;   923          1010  4         lenf = .lenf-1;                     ! Decrement the input string
;   924          1011  4         IF .lenf LSS 0                      ! If we have exhausted the
;   925          1012  4         THEN                                !  filename string then we did not
;   926          1013  4             RETURN false;                   !  match and we can return false
;   927          1014  4         chf = CH$RCHAR_A (adrf);            ! Get filename char and bump pointer
;   928          1015  5         IF NOT (                            ! If none of the successful tests
;   929          1016  6                 (.cht EQL .chf)             ! Did we match?
;   930          1017  6                 OR (.cht EQL '%')           ! Single character wildcard
;   931          1018  5                 )
;   932          1019  4         THEN                                ! We did not match in any way, therefore
;   933          1020  4             RETURN false;                   !  we can return false from here.
;   934          1021  3         END;
;   935          1022  2     END;
;   936          1023  2
;   937          1024  2 IF .lenf EQL 0 THEN RETURN true;            ! Matched so far, filename exhausted
;   938          1025  2
;   939          1026  2 RETURN false;                               ! More characters in filename, no match
;   940          1027  2
;   941          1028  1 END;


                                03FC 00000          .ENTRY  EXCH$CMD_MATCH_FILENAME, Save R2,R3,R4,R5,- ; 0927
                                                            R6,R7,R8,R9
                          52    04   AC  7D 00002          MOVQ    LEN_FIL, LENF                           ; 0978
                          54    0C   AC  7D 00006          MOVQ    LEN_TRG, LENT                           ; 0980
                          59         54  D0 0000A          MOVL    LENT, K                                 ; 0985
                                     36  11 0000D          BRB     5$                                      ;
                                     54  D7 0000F 1$:      DECL    LENT                                    ; 0988
```

```
                                57              85  90 00011          MOVB    (ADRT)+, CHT                            ; 0989
                                2A              57  91 00014          CMPB    CHT, #42                                ; 0991
                                                1B  12 00017          BNEQ    4$
                                                54  D5 00019          TSTL    LENT                                    ; 0994
                                                2F  13 0001B          BEQL    6$
                                58              52  D0 0001D          MOVL    LENF, I                                 ; 1000
                                                0D  11 00020          BRB     3$
                                                3C  BB 00022  2$:     PUSHR   #^M<R2,R3,R4,R5>
                            D8  AF              04  FB 00024          CALLS   #4, EXCH$CMD_MATCH_FILENAME
                                21              50  E8 00028          BLBS    R0, 6$
                                                52  D7 0002B          DECL    LENF                                    ; 1003
                                                53  D6 0002D          INCL    ADRF                                    ; 1004
                                F0              58  F4 0002F  3$:     SOBGEQ  I, 2$                                   ; 0997
                                                1C  11 00032          BRB     7$                                      ; 1006
                                                52  D7 00034  4$:     DECL    LENF                                    ; 1010
                                                18  19 00036          BLSS    7$                                      ; 1011
                                56              83  90 00038          MOVB    (ADRF)+, CHF                            ; 1014
                                56              57  91 0003B          CMPB    CHT, CHF                                ; 1016
                                                05  13 0003E          BEQL    5$
                                25              57  91 00040          CMPB    CHT, #37                                ; 1017
                                                0B  12 00043          BNEQ    7$
                                C7              59  F4 00045  5$:     SOBGEQ  K, 1$                                   ; 0985
                                                52  D5 00048          TSTL    LENF                                    ; 1024
                                                04  12 0004A          BNEQ    7$
                                50              01  D0 0004C  6$:     MOVL    #1, R0
                                                    04 0004F          RET
                                                50  D4 00050  7$:     CLRL    R0                                      ; 1028
                                                    04 00052          RET                                             ;

; Routine Size:  83 bytes,    Routine Base:  EXCH$CMD_CODE + 0649
```

```
;  943     1029  1 GLOBAL ROUTINE exch$cmd_namb_clone (in_namb : $ref_bblock) =    %SBTTL 'cmd_namb_clone (in_namb)'
;  944     1030  2 BEGIN
;  945     1031  2 !++
;  946     1032  2 !
;  947     1033  2 ! FUNCTIONAL DESCRIPTION:
;  948     1034  2 !
;  949     1035  2 !       This routine makes a copy of an EXCHANGE structure $NAMB.
;  950     1036  2 !
;  951     1037  2 ! INPUTS:
;  952     1038  2 !
;  953     1039  2 !       in_namb - address of the created name block
;  954     1040  2 !
;  955     1041  2 ! IMPLICIT INPUTS:
;  956     1042  2 !
;  957     1043  2 !       none
;  958     1044  2 !
;  959     1045  2 ! OUTPUTS:
;  960     1046  2 !
;  961     1047  2 !       none
;  962     1048  2 !
;  963     1049  2 ! IMPLICIT OUTPUTS:
;  964     1050  2 !
;  965     1051  2 !       none
;  966     1052  2 !
;  967     1053  2 ! ROUTINE VALUE:
;  968     1054  2 !
;  969     1055  2 !       Address of a duplicate namb
;  970     1056  2 !
;  971     1057  2 ! SIDE EFFECTS:
;  972     1058  2 !
;  973     1059  2 !       none
;  974     1060  2 !--
;  975     1061  2
;  976     1062  2 $dbgtrc_prefix ('cmd_namb_clone> ');
;  977     1063  2
;  978     1064  2 LOCAL
;  979     1065  2     out_namb    : $ref_bblock                   ! Pointer to the new namb
;  980     1066  2     ;
;  981     1067  2
;  982     1068  2
;  983     1069  2 $block_check (2, .in_namb, namb, 405);
;  984     1070  2
;  985     1071  2 ! Get a new namb
;  986     1072  2 !
;  987     1073  2 out_namb = exch$util_namb_allocate ();
;  988     1074  2
;  989     1075  2 ! Make copies of the dynamic strings
;  990     1076  2 !
;  991     1077  2 str$copy_dx (out_namb [namb$q_input],      in_namb [namb$q_input]);     ! Copy the input name
;  992     1078  2 str$copy_dx (out_namb [namb$q_fullname],   in_namb [namb$q_fullname]);  ! Copy the final, fully expanded nam
;  993     1079  2 str$copy_dx (out_namb [namb$q_expanded],   in_namb [namb$q_expanded]);  ! Make a copy of the expanded name
;  994     1080  2 str$copy_dx (out_namb [namb$q_result],     in_namb [namb$q_result]);    ! Copy the result name
;  995     1081  2 str$copy_dx (out_namb [namb$q_device_dvi], in_namb [namb$q_device_dvi]);! Copy the canonical device name
;  996     1082  2
;  997     1083  2 !+
;  998     1084  2 ! Now we need to get a little smarter.  The namb contains lengths and addresses inside the expanded or resul
;  999     1085  2 ! string for the rest of the filename components.  The safest way is to make new dynamic strings for each of
```

```
;  1000          1086  2 ! these components.  This will result in some small pieces of unrecoverable garbage when this namb is reused
;  1001          1087  2 ! it won't be significant.
;  1002          1088  2 !-
;  1003          1089  2 $dyn_str_desc_init (out_namb [namb$q_node]);              ! Node name descriptor
;  1004          1090  2 str$copy_dx (out_namb [namb$q_node], in_namb [namb$q_node]);
;  1005          1091  2 $dyn_str_desc_init (out_namb [namb$q_device]);            ! Device name descriptor
;  1006          1092  2 str$copy_dx (out_namb [namb$q_device], in_namb [namb$q_device]);
;  1007          1093  2 $dyn_str_desc_init (out_namb [namb$q_directory]);        ! Directory name descriptor
;  1008          1094  2 str$copy_dx (out_namb [namb$q_directory], in_namb [namb$q_directory]);
;  1009          1095  2 $dyn_str_desc_init (out_namb [namb$q_name]);             ! Name name descriptor
;  1010          1096  2 str$copy_dx (out_namb [namb$q_name], in_namb [namb$q_name]);
;  1011          1097  2 $dyn_str_desc_init (out_namb [namb$q_type]);             ! Type name descriptor
;  1012          1098  2 str$copy_dx (out_namb [namb$q_type], in_namb [namb$q_type]);
;  1013          1099  2 $dyn_str_desc_init (out_namb [namb$q_version]);          ! Version name descriptor
;  1014          1100  2 str$copy_dx (out_namb [namb$q_version], in_namb [namb$q_version]);
;  1015          1101  2
;  1016          1102  2 ! Copy the last part of the block exactly
;  1017          1103  2 !
;  1018          1104  2 CH$MOVE (exchblk$s_namb - namb$k_start_zero, .in_namb + namb$k_start_zero, .out_namb + namb$k_start_zero);
;  1019          1105  2
;  1020        P 1106  2 $debug_print_fao (''!AS' '!AS' '!AS' '!AS' '!AS' '!AS' '!AS'',
;  1021        P 1107  2         out_namb [namb$q_node], out_namb [namb$q_device], out_namb [namb$q_directory],
;  1022          1108  2         out_namb [namb$q_name], out_namb [namb$q_type], out_namb [namb$q_version], out_namb [namb$q_device_d
;  1023          1109  2
;  1024          1110  2 RETURN .out_namb;
;  1025          1111  1 END;


                                                .EXTRN  STR$COPY_DX

                                01FC 00000      .ENTRY  EXCH$CMD_NAMB_CLONE, Save R2,R3,R4,R5,R6,-  ; 1029
                                                        R7,R8
                    58 00000000G   EF  9E 00002 MOVAB   TMPL, R8
                    57 00000000G   00  9E 00009 MOVAB   STR$COPY_DX, R7
                    53       04    AC  D0 00010 MOVL    IN_NAMB, R3                                 ; 1069
                    52 010A00F7    8F  D0 00014 MOVL    #17432823, R2
                    51       0195  8F  3C 0001B MOVZWL  #405, R1
                    50             53  D0 00020 MOVL    R3, R0
                           00000000G   EF  16 00023 JSB     EXCH$UTIL_BLOCK_CHECK
              00000000G EF             00  FB 00029 CALLS   #0, EXCH$UTIL_NAMB_ALLOCATE              ; 1073
                    56               50  D0 00030 MOVL    R0, OUT_NAMB
                                 10  A3  9F 00033 PUSHAB  16(R3)                                     ; 1077
                                 10  A6  9F 00036 PUSHAB  16(OUT_NAMB)
                    67               02  FB 00039 CALLS   #2, STR$COPY_DX
                                 18  A3  9F 0003C PUSHAB  24(R3)                                     ; 1078
                                 18  A6  9F 0003F PUSHAB  24(OUT_NAMB)
                    67               02  FB 00042 CALLS   #2, STR$COPY_DX
                                 20  A3  9F 00045 PUSHAB  32(R3)                                     ; 1079
                                 20  A6  9F 00048 PUSHAB  32(OUT_NAMB)
                    67               02  FB 0004B CALLS   #2, STR$COPY_DX
                                 28  A3  9F 0004E PUSHAB  40(R3)                                     ; 1080
                                 28  A6  9F 00051 PUSHAB  40(OUT_NAMB)
                    67               02  FB 00054 CALLS   #2, STR$COPY_DX
                                 30  A3  9F 00057 PUSHAB  48(R3)                                     ; 1081
                                 30  A6  9F 0005A PUSHAB  48(OUT_NAMB)
                    67               02  FB 0005D CALLS   #2, STR$COPY_DX
```

```
                                50        38    A6   9E 00060          MOVAB   56(OUT_NAMB), R0                        ; 1089
                                60              68   7D 00064          MOVQ    TMPL, (R0)
                                          38    A3   9F 00067          PUSHAB  56(R3)                                  ; 1090
                                          38    A6   9F 0006A          PUSHAB  56(OUT_NAMB)
                                67              02   FB 0006D          CALLS   #2, STR$COPY_DX
                                50        40    A6   9E 00070          MOVAB   64(OUT_NAMB), R0                        ; 1091
                                60              68   7D 00074          MOVQ    TMPL, (R0)
                                          40    A3   9F 00077          PUSHAB  64(R3)                                  ; 1092
                                          40    A6   9F 0007A          PUSHAB  64(OUT_NAMB)
                                67              02   FB 0007D          CALLS   #2, STR$COPY_DX
                                50        48    A6   9E 00080          MOVAB   72(OUT_NAMB), R0                        ; 1093
                                60              68   7D 00084          MOVQ    TMPL, (R0)
                                          48    A3   9F 00087          PUSHAB  72(R3)                                  ; 1094
                                          48    A6   9F 0008A          PUSHAB  72(OUT_NAMB)
                                67              02   FB 0008D          CALLS   #2, STR$COPY_DX
                                50        50    A6   9E 00090          MOVAB   80(OUT_NAMB), R0                        ; 1095
                                60              68   7D 00094          MOVQ    TMPL, (R0)
                                          50    A3   9F 00097          PUSHAB  80(R3)                                  ; 1096
                                          50    A6   9F 0009A          PUSHAB  80(OUT_NAMB)
                                67              02   FB 0009D          CALLS   #2, STR$COPY_DX
                                50        58    A6   9E 000A0          MOVAB   88(OUT_NAMB), R0                        ; 1097
                                60              68   7D 000A4          MOVQ    TMPL, (R0)
                                          58    A3   9F 000A7          PUSHAB  88(R3)                                  ; 1098
                                          58    A6   9F 000AA          PUSHAB  88(OUT_NAMB)
                                67              02   FB 000AD          CALLS   #2, STR$COPY_DX
                                50        60    A6   9E 000B0          MOVAB   96(OUT_NAMB), R0                        ; 1099
                                60              68   7D 000B4          MOVQ    TMPL, (R0)
                                          60    A3   9F 000B7          PUSHAB  96(R3)                                  ; 1100
                                          60    A6   9F 000BA          PUSHAB  96(OUT_NAMB)
                                67              02   FB 000BD          CALLS   #2, STR$COPY_DX
                68    A6        68    A3    00A2   8F 28 000C0          MOVC3   #162, 104(R3), 104(OUT_NAMB)            ; 1104
                                50              56   D0 000C8          MOVL    OUT_NAMB, R0                            ; 1110
                                                04 000CB          RET                                             ; 1111

; Routine Size:  204 bytes,    Routine Base:  EXCH$CMD_CODE + 069C
```

```
;  1027     1112  1 GLOBAL ROUTINE cmd_namb_fab_copy (fabb : $ref_bblock, namb : $ref_bblock) : NOVALUE =   %SBTTL 'cmd_namb_fab
;  1028     1113  2 BEGIN
;  1029     1114  2 !++
;  1030     1115  2 !
;  1031     1116  2 ! FUNCTIONAL DESCRIPTION:
;  1032     1117  2 !
;  1033     1118  2 !       This routine copies information from a fab (and its nam block) to a namb.
;  1034     1119  2 !
;  1035     1120  2 ! INPUTS:
;  1036     1121  2 !
;  1037     1122  2 !       fabb - the address of an RMS FAB, assumed to have a nam block
;  1038     1123  2 !       namb - address of the created name block
;  1039     1124  2 !
;  1040     1125  2 ! IMPLICIT INPUTS:
;  1041     1126  2 !
;  1042     1127  2 !       none
;  1043     1128  2 !
;  1044     1129  2 ! OUTPUTS:
;  1045     1130  2 !
;  1046     1131  2 !       none
;  1047     1132  2 !
;  1048     1133  2 ! IMPLICIT OUTPUTS:
;  1049     1134  2 !
;  1050     1135  2 !       none
;  1051     1136  2 !
;  1052     1137  2 ! ROUTINE VALUE:
;  1053     1138  2 !
;  1054     1139  2 !       True if success, warning status code if unable to parse the parameter
;  1055     1140  2 !
;  1056     1141  2 ! SIDE EFFECTS:
;  1057     1142  2 !
;  1058     1143  2 !       $NAMB block is filled with file name information
;  1059     1144  2 !--
;  1060     1145  2
;  1061     1146  2 $dbgtrc_prefix ('cmd_namb_fab_copy> ');
;  1062     1147  2
;  1063     1148  2 LOCAL
;  1064     1149  2     offset,
;  1065     1150  2     len,
;  1066     1151  2     buf,
;  1067     1152  2     tmp_desc    : $desc_block
;  1068     1153  2     ;
;  1069     1154  2
;  1070     1155  2 BIND
;  1071     1156  2     inp_desc = namb [namb$q_input]      : $desc_block,  ! A descriptor for the input file name      (dynamic
;  1072     1157  2     ful_desc = namb [namb$q_fullname]   : $desc_block,  ! A descriptor for the full file name           ..
;  1073     1158  2     exp_desc = namb [namb$q_expanded]   : $desc_block,  ! A descriptor for the expanded file name       ..
;  1074     1159  2     res_desc = namb [namb$q_result]     : $desc_block,  ! A descriptor for the result file name         ..
;  1075     1160  2     dvi_desc = namb [namb$q_device_dvi] : $desc_block,  ! A descriptor for the full device name       ..
;  1076     1161  2     nod_desc = namb [namb$q_node]       : $desc_block,  ! A descriptor for the node name        (point to ex
;  1077     1162  2     dev_desc = namb [namb$q_device]     : $desc_block,  ! A descriptor for the device name            ..
;  1078     1163  2     dir_desc = namb [namb$q_directory]  : $desc_block,  ! A descriptor for the directory name         ..
;  1079     1164  2     nam_desc = namb [namb$q_name]       : $desc_block,  ! A descriptor for the name name              ..
;  1080     1165  2     typ_desc = namb [namb$q_type]       : $desc_block,  ! A descriptor for the type name              ..
;  1081     1166  2     ver_desc = namb [namb$q_version]    : $desc_block,  ! A descriptor for the version number         ..
;  1082     1167  2     nam      = .fabb [fab$l_nam]        : $bblock       ! Address of the name block
;  1083     1168  2     ;
```

```
; 1084        1169  2
; 1085        1170  2 $debug_print_lit ('entry');
; 1086        1171  2 $block_check (2, .namb, namb, 406);
; 1087        1172  2 $logic_check (2, (nam NEQ 0), 103);
; 1088        1173  2
; 1089        1174  2
; 1090        1175  2 ! Copy the DEV characteristics from the fab to the namb
; 1091        1176  2 !
; 1092        1177  2 namb [namb$l_fabdev] = .fabb [fab$l_dev];
; 1093        1178  2
; 1094        1179  2 ! Set some flags based on the file name status bits from the RMS name
; 1095        1180  2 !
; 1096        1181  2 namb [namb$v_wildcard]          = .nam [nam$v_wildcard];        ! Inclusive OR of all the RMS wildcard bits
; 1097        1182  2 namb [namb$v_wild_name]        = .nam [nam$v_wild_name];       ! File name contains a wildcard
; 1098        1183  2 namb [namb$v_wild_type]        = .nam [nam$v_wild_type];       ! File type contains a wildcard
; 1099        1184  2 namb [namb$v_wild_version]     = .nam [nam$v_wild_ver];        ! File version contains a wildcard
; 1100        1185  2 namb [namb$v_wild_group]       = .nam [nam$v_wild_grp];        ! Group number is a wildcard
; 1101        1186  2 namb [namb$v_wild_member]      = .nam [nam$v_wild_mbr];        ! Member number is a wildcard
; 1102        1187  2
; 1103        1188  2 namb [namb$v_rooted_directory] = .nam [nam$v_root_dir];        ! Device contains root directory
; 1104        1189  2 namb [namb$v_concealed_device] = .nam [nam$v_cncl_dev];        ! Device is a concealed device
; 1105        1190  2
; 1106        1191  2 namb [namb$v_explicit_node]    = .nam [nam$v_node];            ! Node name is present
; 1107        1192  2 namb [namb$v_explicit_device]  = .nam [nam$v_exp_dev];         ! Device is explicit
; 1108        1193  2 namb [namb$v_explicit_directory]= .nam [nam$v_exp_dir];        ! Directory is explicit
; 1109        1194  2 namb [namb$v_explicit_name]    = .nam [nam$v_exp_name];        ! Name is explicit
; 1110        1195  2 namb [namb$v_explicit_type]    = .nam [nam$v_exp_type];        ! Type is explicit
; 1111        1196  2 namb [namb$v_explicit_version] = .nam [nam$v_exp_ver];         ! Version number is explicit
; 1112        1197  2
; 1113        1198  2 ! RMS doesn't set the GRP_MBR bit if the device is not directory structured, do it for them
; 1114        1199  2 !
; 1115        1200  2 IF .nam [nam$b_dir] NEQ 0
; 1116        1201  2 THEN
; 1117        1202  2     nam [nam$v_grp_mbr] = (CH$FIND_CH (.nam [nam$b_dir], .nam [nam$l_dir], %C ',') NEQ 0);
; 1118        1203  2 $debug_print_fao ('grp_mbr bit = !OL', .nam [nam$v_grp_mbr]);
; 1119        1204  3 IF (namb [namb$v_uic_directory] = .nam [nam$v_grp_mbr])       ! Directory is in [GROUP,MEMBER] format
; 1120        1205  2 THEN
; 1121        1206  2     cmd_convert_uic (.fabb, .namb);                          ! Also store it in binary format
; 1122        1207  2
; 1123        1208  2 ! Make a copy of the input file name, copy from fab to a dynamic string
; 1124        1209  2 !
; 1125        1210  2 $stat_str_desc_init (tmp_desc, .fabb [fab$b_fns], .fabb [fab$l_fna]);   ! Set class, type, length and pointe
; 1126        1211  2 str$copy_dx (inp_desc, tmp_desc);                               ! Copy to dynamic string
; 1127        1212  2
; 1128        1213  2 ! Make a copy of the expanded name, copy from nam to a dynamic string
; 1129        1214  2 !
; 1130        1215  2 $str_desc_set (tmp_desc, .nam [nam$b_esl], .nam [nam$l_esa]);       ! Set length and pointer fields only
; 1131        1216  2 str$copy_dx (exp_desc, tmp_desc);                               ! Copy to dynamic string
; 1132        1217  2
; 1133        1218  2 ! Make a copy of the result name, copy from nam to a dynamic string
; 1134        1219  2 !
; 1135        1220  2 $str_desc_set (tmp_desc, .nam [nam$b_rsl], .nam [nam$l_rsa]);       ! Set length and pointer fields only
; 1136        1221  2 str$copy_dx (res_desc, tmp_desc);                               ! Copy to dynamic string
; 1137        1222  2
; 1138        1223  2 ! Make a copy of the final name.  This is the result name if present, expanded if not.  If the expanded name
; 1139        1224  2 ! is not present, all bets are off.
; 1140        1225  2 !
```

```
;  1141          1226  3 BEGIN
;  1142          1227  3     REGISTER
;  1143          1228  3         final_len, final_adr;
;  1144          1229  3     IF .nam [nam$b_rsl] GTRU 0
;  1145          1230  3     THEN
;  1146          1231  4         BEGIN
;  1147          1232  4         final_len = .nam [nam$b_rsl];
;  1148          1233  4         final_adr = .nam [nam$l_rsa];
;  1149          1234  4         END
;  1150          1235  3     ELSE IF .nam [nam$b_esl] GTRU 0
;  1151          1236  3     THEN
;  1152          1237  4         BEGIN
;  1153          1238  4         final_len = .nam [nam$b_esl];
;  1154          1239  4         final_adr = .nam [nam$l_esa];
;  1155          1240  4         END
;  1156          1241  3     ELSE
;  1157          1242  3         $logic_check (0, (false), 232);
;  1158          1243  3
;  1159          1244  3     $str_desc_set (tmp_desc, .final_len, .final_adr);   ! Set length and pointer fields only
;  1160          1245  2     END;
;  1161          1246  2 str$copy_dx (ful_desc, tmp_desc);                     ! Copy to dynamic string
;  1162          1247  2
;  1163          1248  2 ! Make a copy of the canonical device name, copy from nam to a dynamic string
;  1164          1249  2 !
;  1165          1250  3 BEGIN                                                 ! The DVI field is a counted string, so we
;  1166          1251  3     BIND                                              !  must trick BLISS into thinking the first
;  1167          1252  3         len = nam [nam$t_dvi]     : BYTE,             !  byte is the length field and the second
;  1168          1253  3         adr = nam [nam$t_dvi] + 1 : $bvector;         !  byte is the string itself
;  1169          1254  3     $str_desc_set (tmp_desc, .len, adr);              ! Set length and pointer fields only
;  1170          1255  2 END;
;  1171          1256  2 str$copy_dx (dvi_desc, tmp_desc);                     ! Copy to dynamic string
;  1172          1257  2
;  1173          1258  2 !+
;  1174          1259  2 ! Now we need to get a little smarter.  The nam block contains lengths and addresses inside the expanded or
;  1175          1260  2 ! string for the rest of the filename components.  Since we have made a copy of this string, we must convert
;  1176          1261  2 ! nam block addresses to addresses inside our copy.  This is a simple matter of arithmetic.  We must also
;  1177          1262  2 ! initialize all these as static strings, so here we go:
;  1178          1263  2 !-
;  1179          1264  2 IF .nam [nam$b_rsl] GTRU 0                           ! Is the result string available?
;  1180          1265  2 THEN
;  1181          1266  2     offset = .res_desc [dsc$a_pointer] - .nam [nam$l_rsa]       ! Compute our offset
;  1182          1267  2 ELSE
;  1183          1268  2     offset = .exp_desc [dsc$a_pointer] - .nam [nam$l_esa];      ! Compute our offset
;  1184          1269  2
;  1185          1270  2 $stat_str_desc_init (nod_desc, .nam [nam$b_node], (.nam [nam$l_node] + .offset));       ! Node name descript
;  1186          1271  2 $stat_str_desc_init (dev_desc, .nam [nam$b_dev],  (.nam [nam$l_dev]  + .offset));       ! Device name descri
;  1187          1272  2 $stat_str_desc_init (dir_desc, .nam [nam$b_dir],  (.nam [nam$l_dir]  + .offset));       ! Directory name des
;  1188          1273  2 $stat_str_desc_init (nam_desc, .nam [nam$b_name], (.nam [nam$l_name] + .offset));       ! Name name descript
;  1189          1274  2 $stat_str_desc_init (typ_desc, .nam [nam$b_type], (.nam [nam$l_type] + .offset));       ! Type name descript
;  1190          1275  2 $stat_str_desc_init (ver_desc, .nam [nam$b_ver],  (.nam [nam$l_ver]  + .offset));       ! Version name descr
;  1191          1276  2
;  1192          1277  2 ! Check now to make sure that we can use this as one of our foreign names, look first at the length
;  1193          1278  2 !
;  1194          1279  2 IF   .nam [nam$b_type] GTRU 4                        ! If the type is longer than three (includes the ".")
;  1195          1280  2   OR
;  1196          1281  2      .nam [nam$b_name] GTRU 9                        !  or the name is longer than 9
;  1197          1282  2 THEN
```

```
;  1198          1283  3      BEGIN                                   !  then we need to truncate DOS and RT output names
;  1199          1284  3      namb [namb$v_dos_truncate] = true;
;  1200          1285  3      namb [namb$v_rt_truncate] = true;
;  1201          1286  3      END
;  1202          1287  2  ELSE IF .nam [nam$b_name] GTRU 6              ! If the name is longer than 6 we need to truncate RT
;  1203          1288  2  THEN
;  1204          1289  2      namb [namb$v_rt_truncate] = true;
;  1205          1290  2
;  1206          1291  2  ! Now look for characters which cannot be put into the other file names
;  1207          1292  2  !
;  1208          1293  2  len = .nam [nam$b_type] - 1;                  ! Do the file type first, adjust for the "."
;  1209          1294  2  buf = .nam [nam$l_type] + 1;
;  1210          1295  2  WHILE .len GTR 0
;  1211          1296  2  DO
;  1212          1297  3      BEGIN
;  1213          1298  3      REGISTER
;  1214          1299  3          char : BYTE;
;  1215          1300  3      char = CH$RCHAR_A (buf);
;  1216          1301  3      SELECTONE .char OF
;  1217          1302  3      SET
;  1218          1303  3          ['A' TO 'Z', '0' TO '9', '$', '%'] :    ;                          ! Valid chars plus wildcards
;  1219          1304  3          [OTHERWISE] :                           namb [namb$v_bad_pdp_char] = true;
;  1220          1305  3      TES;
;  1221          1306  3      len = .len - 1;
;  1222          1307  3      END;
;  1223          1308  2
;  1224          1309  2  len = .nam [nam$b_name];                      ! Do the file name next
;  1225          1310  2  buf = .nam [nam$l_name];
;  1226          1311  2  WHILE .len GTR 0
;  1227          1312  2  DO
;  1228          1313  3      BEGIN
;  1229          1314  3      REGISTER
;  1230          1315  3          char : BYTE;
;  1231          1316  3      char = CH$RCHAR_A (buf);
;  1232          1317  3      SELECTONE .char OF
;  1233          1318  3      SET
;  1234          1319  3          ['A' TO 'Z', '0' TO '9', '$', '%'] :    ;                          ! Valid chars plus wildcards
;  1235          1320  3          [OTHERWISE] :                           namb [namb$v_bad_pdp_char] = true;
;  1236          1321  3      TES;
;  1237          1322  3      len = .len - 1;
;  1238          1323  3      END;
;  1239          1324  2
;  1240        P 1325  2  $debug_print_fao ('"!AS" "!AS" "!AS" "!AS" "!AS" "!AS" "!AS"',
;  1241          1326  2          nod_desc, dev_desc, dir_desc, nam_desc, typ_desc, ver_desc, dvi_desc);
;  1242          1327  2
;  1243          1328  2  RETURN;
;  1244          1329  1  END;


                                  0FFC 00000          .ENTRY  CMD_NAMB_FAB_COPY, Save R2,R3,R4,R5,R6,R7,- ; 1112
                                                              R8,R9,R10,R11                                 ;
                5B 00000000G   8F  DD 00002           MOVL    #EXCH$_BADLOGIC, R11                        ;
                5A 00000000G   00  9E 00009           MOVAB   STR$COPY_DX, R10                            ;
                5E             08  C2 00010           SUBL2   #8, SP                                      ;
```

```
                                55        08    AC  D0 00013          MOVL    NAMB, R5                                        ; 1156
                                58        20    A5  9E 00017          MOVAB   32(R5), R8                                      ; 1158
                                57        28    A5  9E 0001B          MOVAB   40(R5), R7                                      ; 1159
                                53        04    AC  D0 0001F          MOVL    FABB, R3                                        ; 1167
                                54        28    A3  D0 00023          MOVL    40(R3), R4                                      ;
                                52  010A00F7    8F  D0 00027          MOVL    #17432823, R2                                   ; 1171
                                51      0196    8F  3C 0002E          MOVZWL  #406, R1                                        ;
                                50              55  D0 00033          MOVL    R5, R0                                          ;
                                    00000000G   EF  16 00036          JSB     EXCH$UTIL_BLOCK_CHECK                           ;
                                                54  D5 0003C          TSTL    R4                                              ; 1172
                                                0F  12 0003E          BNEQ    1$                                              ;
                                7E        67    8F  9A 00040          MOVZBL  #103, -(SP)                                     ;
                                                01  DD 00044          PUSHL   #1                                              ;
                                                5B  DD 00046          PUSHL   R11                                             ;
                      00000000G 00              03  FB 00048          CALLS   #3, LIB$STOP                                    ;
                            68  A5        40    A3  D0 0004F  1$:     MOVL    64(R3), 104(R5)                                 ; 1177
                                56        6C    A5  9E 00054          MOVAB   108(R5), R6                                     ; 1181
                                52        34    A4  9E 00058          MOVAB   52(R4), R2                                      ;
      66                        00        01    A2  F0 0005C          INSV    1(R2), #0, #1, (R6)                             ;
      50          62            01              05  EF 00062          EXTZV   #5, #1, (R2), R0                                ; 1182
      66          01            01              50  F0 00067          INSV    R0, #1, #1, (R6)                                ;
      50          62            01              04  EF 0006C          EXTZV   #4, #1, (R2), R0                                ; 1183
      66          01            02              50  F0 00071          INSV    R0, #2, #1, (R6)                                ;
      50          62            01              03  EF 00076          EXTZV   #3, #1, (R2), R0                                ; 1184
      66          01            03              50  F0 0007B          INSV    R0, #3, #1, (R6)                                ;
      66          01            04        03    A2  F0 00080          INSV    3(R2), #4, #1, (R6)                             ; 1185
      50          62            01              19  EF 00086          EXTZV   #25, #1, (R2), R0                               ; 1186
      66          01            05              50  F0 0008B          INSV    R0, #5, #1, (R6)                                ;
      50          62            01              0D  EF 00090          EXTZV   #13, #1, (R2), R0                               ; 1188
      66          01            0D              50  F0 00095          INSV    R0, #13, #1, (R6)                               ;
      50          62            01              0C  EF 0009A          EXTZV   #12, #1, (R2), R0                               ; 1189
      66          01            0C              50  F0 0009F          INSV    R0, #12, #1, (R6)                               ;
      50          62            01              11  EF 000A4          EXTZV   #17, #1, (R2), R0                               ; 1191
      66          01            06              50  F0 000A9          INSV    R0, #6, #1, (R6)                                ;
      50          62            01              07  EF 000AE          EXTZV   #7, #1, (R2), R0                                ; 1192
      66          01            07              50  F0 000B3          INSV    R0, #7, #1, (R6)                                ;
      50          62            01              06  EF 000B8          EXTZV   #6, #1, (R2), R0                                ; 1193
01    A6          01            00              50  F0 000BD          INSV    R0, #0, #1, 1(R6)                               ;
      50          62            01              02  EF 000C3          EXTZV   #2, #1, (R2), R0                                ; 1194
      66          01            09              50  F0 000C8          INSV    R0, #9, #1, (R6)                                ;
      50          62            01              01  EF 000CD          EXTZV   #1, #1, (R2), R0                                ; 1195
      66          01            0A              50  F0 000D2          INSV    R0, #10, #1, (R6)                               ;
      66          01            0B              62  F0 000D7          INSV    (R2), #11, #1, (R6)                             ; 1196
                                          3A    A4  95 000DC          TSTB    58(R4)                                          ; 1200
                                                1A  13 000DF          BEQL    4$                                              ;
                                50        3A    A4  9A 000E1          MOVZBL  58(R4), R0                                      ; 1202
                  48    B4      50              2C  3A 000E5          LOCC    #44, R0, @72(R4)                                ;
                                                02  12 000EA          BNEQ    2$                                              ;
                                                51  D4 000EC          CLRL    R1                                              ;
                                                50  D4 000EE  2$:     CLRL    R0                                              ;
                                                51  D5 000F0          TSTL    R1                                              ;
                                                02  13 000F2          BEQL    3$                                              ;
                                                50  D6 000F4          INCL    R0                                              ;
      62          01            13              50  F0 000F6  3$:     INSV    R0, #19, #1, (R2)                               ;
      52          62            01              13  EF 000FB  4$:     EXTZV   #19, #1, (R2), R2                               ; 1204
      66          01            0E              52  F0 00100          INSV    R2, #14, #1, (R6)                               ;
                                07              52  E9 00105          BLBC    R2, 5$                                          ;
```

```
                                 28 BB 00108          PUSHR   #^M<R3,R5>                          ; 1206
                           F83C  CF                   02 FB 0010A          CALLS   #2, CMD_CONVERT_UIC
                             02  AE          010E     8F B0 0010F 5$:      MOVW    #270, DESC+2        ; 1210
                                 6E            34     A3 9B 00115          MOVZBW  52(R3), DESC
                             04  AE            2C     A3 D0 00119          MOVL    44(R3), DESC+4
                                 5E                   DD 0011E             PUSHL   SP                  ; 1211
                                               10     A5 9F 00120          PUSHAB  16(R5)
                                 6A                   02 FB 00123          CALLS   #2, STR$COPY_DX
                                 6E            0B     A4 9B 00126          MOVZBW  11(R4), DESC        ; 1215
                             04  AE            0C     A4 D0 0012A          MOVL    12(R4), DESC+4
                                           4100     8F BB 0012F          PUSHR   #^M<R8,SP>          ; 1216
                                 6A                   02 FB 00133          CALLS   #2, STR$COPY_DX
                                 6E            03     A4 9B 00136          MOVZBW  3(R4), DESC         ; 1220
                             04  AE            04     A4 D0 0013A          MOVL    4(R4), DESC+4
                                           4080     8F BB 0013F          PUSHR   #^M<R7,SP>          ; 1221
                                 6A                   02 FB 00143          CALLS   #2, STR$COPY_DX
                                                      59 D4 00146          CLRL    R9                  ; 1229
                                               03     A4 95 00148          TSTB    3(R4)
                                                      0C 13 0014B          BEQL    6$
                                                      59 D6 0014D          INCL    R9
                                 52            03     A4 9A 0014F          MOVZBL  3(R4), FINAL_LEN    ; 1232
                                 53            04     A4 D0 00153          MOVL    4(R4), FINAL_ADR    ; 1233
                                                      1E 11 00157          BRB     8$                  ; 1229
                                               0B     A4 95 00159 6$:      TSTB    11(R4)              ; 1235
                                                      0A 13 0015C          BEQL    7$
                                 52            0B     A4 9A 0015E          MOVZBL  11(R4), FINAL_LEN   ; 1238
                                 53            0C     A4 D0 00162          MOVL    12(R4), FINAL_ADR   ; 1239
                                                      0F 11 00166          BRB     8$                  ; 1235
                                 7E            E8     8F 9A 00168 7$:      MOVZBL  #232, -(SP)         ; 1242
                                                      01 DD 0016C          PUSHL   #1
                                                      5B DD 0016E          PUSHL   R11
                       00000000G  00                  03 FB 00170          CALLS   #3, LIB$STOP
                                 6E                   52 B0 00177 8$:      MOVW    FINAL_LEN, DESC     ; 1244
                             04  AE                   53 D0 0017A          MOVL    FINAL_ADR, DESC+4
                                 5E                   DD 0017E             PUSHL   SP                  ; 1246
                                               18     A5 9F 00180          PUSHAB  24(R5)
                                 6A                   02 FB 00183          CALLS   #2, STR$COPY_DX
                                 6E            14     A4 9B 00186          MOVZBW  20(R4), DESC        ; 1254
                             04  AE            15     A4 9E 0018A          MOVAB   21(R4), DESC+4
                                 5E                   DD 0018F             PUSHL   SP                  ; 1256
                                               30     A5 9F 00191          PUSHAB  48(R5)
                                 6A                   02 FB 00194          CALLS   #2, STR$COPY_DX
                                 08                   59 E9 00197          BLBC    R9, 9$              ; 1264
                         51      04  A7            04     A4 C3 0019A          SUBL3   4(R4), 4(R7), OFFSET ; 1266
                                                      06 11 001A0          BRB     10$
                         51      04  A8            0C     A4 C3 001A2 9$:      SUBL3   12(R4), 4(R8), OFFSET ; 1268
                                 50            38     A5 9E 001A8 10$:     MOVAB   56(R5), R0          ; 1270
                             02  A0          010E     8F B0 001AC          MOVW    #270, 2(R0)
                                 60            38     A4 9B 001B2          MOVZBW  56(R4), (R0)
                             04  A0            40 B441  9E 001B6          MOVAB   @64(R4)[OFFSET], 4(R0)
                                 50            40     A5 9E 001BC          MOVAB   64(R5), R0          ; 1271
                             02  A0          010E     8F B0 001C0          MOVW    #270, 2(R0)
                                 60            39     A4 9B 001C6          MOVZBW  57(R4), (R0)
                             04  A0            44 B441  9E 001CA          MOVAB   @68(R4)[OFFSET], 4(R0)
                                 50            48     A5 9E 001D0          MOVAB   72(R5), R0          ; 1272
                             02  A0          010E     8F B0 001D4          MOVW    #270, 2(R0)
                                 60            3A     A4 9B 001DA          MOVZBW  58(R4), (R0)
```

```
                    04   A0      48 B441   9E 001DE          MOVAB   @72(R4)[OFFSET], 4(R0)
                    50    50       A5   9E 001E4          MOVAB   80(R5), R0                      ;1273
                    02   A0    010E   8F   B0 001E8          MOVW    #270, 2(R0)
                    60    3B       A4   9B 001EE          MOVZBW  59(R4), (R0)
                    04   A0      4C B441   9E 001F2          MOVAB   @76(R4)[OFFSET], 4(R0)
                    50    58       A5   9E 001F8          MOVAB   88(R5), R0                      ;1274
                    02   A0    010E   8F   B0 001FC          MOVW    #270, 2(R0)
                    60    3C       A4   9B 00202          MOVZBW  60(R4), (R0)
                    04   A0      50 B441   9E 00206          MOVAB   @80(R4)[OFFSET], 4(R0)
                    50    60       A5   9E 0020C          MOVAB   96(R5), R0                      ;1275
                    02   A0    010E   8F   B0 00210          MOVW    #270, 2(R0)
                    60    3D       A4   9B 00216          MOVZBW  61(R4), (R0)
                    04   A0      54 B441   9E 0021A          MOVAB   @84(R4)[OFFSET], 4(R0)
                          04       3C   A4   91 00220          CMPB    60(R4), #4              ;1279
                                                06   1A 00224          BGTRU   11$
                          09       3B   A4   91 00226          CMPB    59(R4), #9              ;1281
                                                06   1B 0022A          BLEQU   12$
                    02   A6                   02   88 0022C 11$:      BISB2   #2, 2(R6)       ;1284
                                                06   11 00230          BRB     13$            ;1285
                          06       3B   A4   91 00232 12$:      CMPB    59(R4), #6              ;1287
                                                04   1B 00236          BLEQU   14$
                    02   A6                   04   88 00238 13$:      BISB2   #4, 2(R6)       ;1289
                          51       3C   A4   9A 0023C 14$:      MOVZBL  60(R4), LEN             ;1293
                                                51   D7 00240          DECL    LEN
                52        50   A4              01   C1 00242          ADDL3   #1, 80(R4), BUF    ;1294
                                                51   D5 00247 15$:      TSTL    LEN            ;1295
                                                2B   15 00249          BLEQ    19$
                          50                    82   90 0024B          MOVB    (BUF)+, CHAR   ;1300
                          25                    50   91 0024E          CMPB    CHAR, #37      ;1303
                                                1F   13 00251          BEQL    18$
                          2A                    50   91 00253          CMPB    CHAR, #42
                                                1A   13 00256          BEQL    18$
                          30                    50   91 00258          CMPB    CHAR, #48
                                                05   1F 0025B          BLSSU   16$
                          39                    50   91 0025D          CMPB    CHAR, #57
                                                10   1B 00260          BLEQU   18$
                41        8F                    50   91 00262 16$:      CMPB    CHAR, #65
                                                06   1F 00266          BLSSU   17$
                5A        8F                    50   91 00268          CMPB    CHAR, #90
                                                04   1B 0026C          BLEQU   18$
                02        A6                    01   88 0026E 17$:      BISB2   #1, 2(R6)      ;1304
                                                51   D7 00272 18$:      DECL    LEN            ;1306
                                                D1   11 00274          BRB     15$            ;1295
                          51       3B   A4   9A 00276 19$:      MOVZBL  59(R4), LEN            ;1309
                          52       4C   A4   D0 0027A          MOVL    76(R4), BUF            ;1310
                                                51   D5 0027E 20$:      TSTL    LEN            ;1311
                                                2B   15 00280          BLEQ    24$
                          50                    82   90 00282          MOVB    (BUF)+, CHAR   ;1316
                          25                    50   91 00285          CMPB    CHAR, #37      ;1319
                                                1F   13 00288          BEQL    23$
                          2A                    50   91 0028A          CMPB    CHAR, #42
                                                1A   13 0028D          BEQL    23$
                          30                    50   91 0028F          CMPB    CHAR, #48
                                                05   1F 00292          BLSSU   21$
                          39                    50   91 00294          CMPB    CHAR, #57
                                                10   1B 00297          BLEQU   23$
                41        8F                    50   91 00299 21$:      CMPB    CHAR, #65
```

```
                                 06  1F 0029D          BLSSU   22$                                    ;
                          5A  8F 50  91 0029F          CMPB    CHAR, #90                              ;
                                 04  1B 002A3          BLEQU   23$                                    ;
                          02  A6 01  88 002A5 22$:      BISB2   #1, 2(R6)                              ; 1320
                                 51  D7 002A9 23$:      DECL    LEN                                    ; 1322
                                 D1  11 002AB          BRB     20$                                    ; 1311
                                     04 002AD 24$:      RET                                            ; 1329

; Routine Size:  686 bytes,    Routine Base:  EXCH$CMD_CODE + 0768
```

```
;  1246          1330  1 GLOBAL ROUTINE exch$cmd_parse_filespec (para_name : $ref_bblock,        %SBTTL 'exch$cmd_parse_filespec'
;  1247          1331  1                                       default_desc : $ref_bblock, prsopt : $bblock, name_desc : $ref_bblock, namb : $ref_b
;  1248          1332  2 BEGIN
;  1249          1333  2 !++
;  1250          1334  2 !
;  1251          1335  2 ! FUNCTIONAL DESCRIPTION:
;  1252          1336  2 !
;  1253          1337  2 !       This routine parses a file name parameter into its component parts.  The output is a filled name blo
;  1254          1338  2 !
;  1255          1339  2 ! INPUTS:
;  1256          1340  2 !
;  1257          1341  2 !       para_name     - the descriptor for the name of the parameter to be given to CLI$GET_VALUE
;  1258          1342  2 !       default_desc - a default name to supply fields missing in the above name.  File name stickiness
;  1259          1343  2 !                       can be achieved by using a previously returned name as the new default.
;  1260          1344  2 !       prsopt        - flags for parse options
;  1261          1345  2 !
;  1262          1346  2 ! IMPLICIT INPUTS:
;  1263          1347  2 !
;  1264          1348  2 !       none
;  1265          1349  2 !
;  1266          1350  2 ! OUTPUTS:
;  1267          1351  2 !
;  1268          1352  2 !       name_desc - the address of a dynamic string descriptor, will receive the name from CLI$GET_VALUE
;  1269          1353  2 !       namb      - address of the created name block
;  1270          1354  2 !
;  1271          1355  2 ! IMPLICIT OUTPUTS:
;  1272          1356  2 !
;  1273          1357  2 !       none
;  1274          1358  2 !
;  1275          1359  2 ! ROUTINE VALUE:
;  1276          1360  2 !
;  1277          1361  2 !       True if success, bad status code if unable to parse the parameter, zero if no more parameters
;  1278          1362  2 !
;  1279          1363  2 ! SIDE EFFECTS:
;  1280          1364  2 !
;  1281          1365  2 !       none
;  1282          1366  2 !--
;  1283          1367  2
;  1284          1368  2 $dbgtrc_prefix ('cmd_parse_filespec> ');
;  1285          1369  2
;  1286          1370  2 LOCAL
;  1287          1371  2     fab          : $bblock [fab$k_bln],          ! FAB for the RMS parse
;  1288          1372  2     nam          : $bblock [nam$k_bln],          ! NAM for the RMS parse
;  1289          1373  2     ebuf         : $bblock [nam$c_maxrss],       ! Expanded name buffer for the RMS parse
;  1290          1374  2     namb_ptr     : $ref_bblock,                  ! Local pointer for the newly created block
;  1291          1375  2     cli_status,                                  ! Status from CLI$GET_VALUE
;  1292          1376  2     prs_status,                                  ! Status from RMS parse
;  1293          1377  2     status,                                      ! Status from other things
;  1294          1378  2     dev_item     : VECTOR [10, LONG],            ! Item list for $GETDVI
;  1295          1379  2     dev_desc     : $desc_block,                  ! Name string for $GETDVI
;  1296          1380  2     devnam       : $bvector [16],                ! Name buffer
;  1297          1381  2     devnamlen    : WORD                          ! Returned length of name
;  1298          1382  2     ;
;  1299          1383  2
;  1300          1384  2 BIND
;  1301          1385  2     nam_dvilen = nam [nam$t_dvi] : BYTE,         ! DVI name is counted string
;  1302          1386  2     nam_dvibuf = nam [nam$t_dvi]+1 : $bvector
```

```
;  1303          1387  2     ;
;  1304          1388  2
;  1305          1389  2 $trace_print_lit ('entry');
;  1306          1390  2
;  1307          1391  2 ! Set the returned name structure to the null pointer
;  1308          1392  2 !
;  1309          1393  2 .namb = 0;
;  1310          1394  2
;  1311          1395  2 ! Get the string from the CLI.  Note that we must get the parameter in this routine so that our later calls
;  1312          1396  2 ! get the values of /RECORD_FORMAT and /VOLUME_FORMAT will get the correct placement (local).
;  1313          1397  2 !
;  1314          1398  2 IF .prsopt [prsopt$v_no_get_value]                ! If the no_get_value option is requested, simply copy the
;  1315          1399  2 THEN                                              !  input parameter name to the output string.  Note that we
;  1316          1400  2     str$copy_dx (.name_desc, .para_name)           !  can not guarantee proper /RECORD and /VOLUME placement.
;  1317          1401  2 ELSE
;  1318          1402  2     BEGIN
;  1319          1403  3     IF NOT (cli_status = cli$get_value (.para_name, .name_desc))
;  1320          1404  3     THEN
;  1321          1405  4         RETURN 0;
;  1322          1406  3     END;
;  1323          1407  2
;  1324          1408  2 ! Initialize the structures needed by RMS PARSE routine
;  1325          1409  2 !
;  1326        P 1410  2 $fab_init (
;  1327        P 1411  2           fab = fab,                              ! Input file FAB
;  1328        P 1412  2           fna = .name_desc [dsc$a_pointer],       !  Set name addr
;  1329        P 1413  2           fns = .name_desc [dsc$w_length],        !  Set name size
;  1330          1414  2           nam = nam);                             !  Name block
;  1331          1415  2
;  1332        P 1416  2 $nam_init (
;  1333        P 1417  2           nam = nam,                              ! File name block
;  1334        P 1418  2           esa = ebuf,                             !  Expanded name buffer addr
;  1335          1419  2           ess = nam$c_maxrss);                    !  Expanded name size
;  1336          1420  2 !?        nop = synchk);                          !  Syntax check only (needs some more work to use)
;  1337          1421  2
;  1338          1422  2 ! If a default name has been supplied, pass it on to the parse routine as the default
;  1339          1423  2 !
;  1340          1424  2 IF .default_desc NEQ 0
;  1341          1425  2 THEN
;  1342          1426  3     BEGIN
;  1343          1427  3     fab [fab$b_dns] = .default_desc [dsc$w_length];
;  1344          1428  3     fab [fab$l_dna] = .default_desc [dsc$a_pointer];
;  1345          1429  3     END;
;  1346          1430  2
;  1347          1431  2 ! Now let RMS do the dirty work for us
;  1348          1432  2 !
;  1349          1433  2 $debug_print_lit ('fab before parse   <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<');
;  1350          1434  2 $check_call (5, exch$dbg_fab_dump, fab);          ! Dump the fab
;  1351          1435  2 prs_status = $parse (FAB = fab);                  ! Parse the file spec
;  1352          1436  2 exch$cmd_parse_null_file (fab);                   ! And parse the null
;  1353          1437  2 $trace_print_lit ('fab after parse    >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>');
;  1354          1438  2 $check_call (4, exch$dbg_fab_dump, fab);          ! Dump the fab
;  1355          1439  2 $check_call (4, exch$dbg_fab_name_print, fab);    ! Print name block information
;  1356          1440  2 $check_call (4, exch$dbg_fab_dev_print, fab);     ! Print device information
;  1357          1441  2
;  1358          1442  2 ! If we don't have an expanded name we exit with the error
;  1359          1443  2 !
```

```
;  1360     1444  2 IF .nam [nam$b_esl] EQL 0
;  1361     1445  2 THEN
;  1362     1446  2     RETURN .prs_status;
;  1363     1447  2
;  1364     1448  2 !
;  1365     1449  2 ! If we got an RMS$_DNR (device not ready or not mounted) error, then the device exists.  Get its real name
;  1366     1450  2 ! and put the real name into the DVI field of the nam block.  Also grab the device characteristics and stick
;  1367     1451  2 ! the fab.
;  1368     1452  2 !
;  1369     1453  2 IF .prs_status EQL rms$_dnr
;  1370     1454  2 THEN
;  1371     1455  2     BEGIN
;  1372     1456  2
;  1373     1457  2     ! Initialize the device name descriptor from the result name
;  1374     1458  2     !
;  1375     1459  2     $logic_check (2, (.nam [nam$b_dev] NEQ 0), 104);
;  1376     1460  2     $stat_str_desc_init (dev_desc, .nam [nam$b_dev],  .nam [nam$l_dev]);
;  1377     1461  2
;  1378     1462  2     ! Initialize the item list for the $GETDVI
;  1379     1463  2     !
;  1380     1464  2     %IF switch_variant GEQ 3 %THEN devnamlen = 0; %FI   ! Suppress uninit reference message while debugging
;  1381     1465  2     dev_item [0] = (dvi$_fulldevnam^16 OR 16);  ! Canonical device name (Note: RMS uses FULLDEVNAM in $parse
;  1382     1466  2     dev_item [1] = devnam;                      ! Buffer for name
;  1383     1467  2     dev_item [2] = devnamlen;                   ! Returned length
;  1384     1468  2     dev_item [3] = (dvi$_devchar^16 OR 4);      ! Device characteristics
;  1385     1469  2     dev_item [4] = fab [fab$l_dev];             ! Plop it right into the fab
;  1386     1470  2     dev_item [5] = 0;                           ! Don't need returned length
;  1387     1471  2     dev_item [6] = 0;                           ! End of list
;  1388     1472  2
;  1389     1473  2     ! Get the device information
;  1390     1474  2     !
;  1391     1475  3     IF NOT (status = $getdviw (efn=0, devnam=dev_desc, itmlst=dev_item))
;  1392     1476  3     THEN
;  1393     1477  3         $exch_signal_stop (exch$_accessfail, 1, dev_desc, .status);
;  1394     1478  3
;  1395     1479  3     ! Copy the name to the nam block, it will look like $PARSE put it there
;  1396     1480  3     !
;  1397     1481  3     $logic_check (2, (.devnamlen LEQU nam$s_dvi-1), 105);
;  1398     1482  3     nam_dvilen = .devnamlen - 1;                ! Copy the length to the counted string
;  1399     1483  3     CH$MOVE (.nam_dvilen, devnam, nam_dvibuf);  ! And copy the bytes (minus the final ":")
;  1400     1484  3     $trace_print_fao ('dnr, fetched device "!AF"', .nam_dvilen, devnam);
;  1401     1485  3
;  1402     1486  3     END;
;  1403     1487  2
;  1404     1488  2 ! Get a pointer to a name block
;  1405     1489  2 !
;  1406     1490  2 namb_ptr = exch$util_namb_allocate ();
;  1407     1491  2
;  1408     1492  2 ! Now copy everything over to the name block and return the pointer
;  1409     1493  2 !
;  1410     1494  2 cmd_namb_fab_copy (fab, .namb_ptr);
;  1411     1495  2
;  1412   P 1496  2 $trace_print_fao ('Namb for input "!AS", full "!AS", device "!AS"',
;  1413     1497  2         namb_ptr [namb$q_input], namb_ptr [namb$q_fullname], namb_ptr [namb$q_device_dvi]);
;  1414     1498  2
;  1415     1499  3 BEGIN    ! scope "devdvidsc"
;  1416     1500  3     BIND
```

```
;  1417          1501  3          devdvidsc = namb_ptr [namb$q_device_dvi] : $desc_block;
;  1418          1502  3
;  1419          1503  3      IF .devdvidsc [dsc$w_length] GTR 0
;  1420          1504  3      THEN
;  1421          1505  4          BEGIN
;  1422          1506  4          LOCAL
;  1423          1507  4              devclass,
;  1424          1508  4              devtype;
;  1425          1509  4
;  1426          1510  4          ! Init the item list for a $GETDVI to get the device class
;  1427          1511  4          !
;  1428          1512  4          %IF switch_variant GEQ 3 %THEN devclass = devtype = 0; %FI      ! Suppress uninit reference message
;  1429          1513  4          dev_item [0] = (dvi$_devclass^16 OR 4);          ! Device class (a DC$_xxx symbol)
;  1430          1514  4          dev_item [1] = devclass;                         ! Buffer for value
;  1431          1515  4          dev_item [2] = 0;                                ! Returned length not important
;  1432          1516  4          dev_item [3] = (dvi$_devtype^16 OR 4);           ! Device type (a DT$_xxx symbol)
;  1433          1517  4          dev_item [4] = devtype;
;  1434          1518  4          dev_item [5] = 0;
;  1435          1519  4          dev_item [6] = (dvi$_devchar^16 OR 4);           ! Device characteristics
;  1436          1520  4          dev_item [7] = namb_ptr [namb$l_fabdev];         ! Plop it right into the namb
;  1437          1521  4          dev_item [8] = 0;
;  1438          1522  4          dev_item [9] = 0;                                ! End of list
;  1439          1523  4
;  1440          1524  4          ! Get the device information
;  1441          1525  4          !
;  1442          1526  5          IF NOT (status = $getdviw (efn=0, devnam=namb_ptr [namb$q_device_dvi], itmlst=dev_item))
;  1443          1527  4          THEN
;  1444          1528  4              $exch_signal_stop (exch$_accessfail, 1, namb_ptr [namb$q_device_dvi], .status);
;  1445          1529  4          namb_ptr [namb$b_devclass] = .devclass; ! Store the value into the namb
;  1446          1530  4          namb_ptr [namb$b_devtype] = .devtype;   ! Store the value into the namb
;  1447          1531  4
;  1448          1532  4          $trace_print_fao ('class !UL, type !UL', .devclass, .devtype);
;  1449          1533  3          END;
;  1450          1534  2      END;    ! scope "devdvilen"
;  1451          1535  2
;  1452          1536  2      ! Grab volume and record format info.  Note that we will get positional or local values for this parameter o
;  1453          1537  2      !
;  1454          1538  3      IF NOT (status = exch$cmd_fetch_rec_format (.namb_ptr))
;  1455          1539  2      THEN
;  1456          1540  3          BEGIN
;  1457          1541  3          exch$util_namb_release (.namb_ptr);
;  1458          1542  3          RETURN .status;
;  1459          1543  3          END;
;  1460          1544  2
;  1461          1545  3      IF NOT (status = exch$cmd_fetch_vol_format (.namb_ptr))
;  1462          1546  2      THEN
;  1463          1547  3          BEGIN
;  1464          1548  3          exch$util_namb_release (.namb_ptr);
;  1465          1549  3          RETURN .status;
;  1466          1550  3          END;
;  1467          1551  2
;  1468          1552  2      ! Produce an ident string for the volume
;  1469          1553  2      !
;  1470          1554  3      BEGIN
;  1471          1555  3          LOCAL
;  1472          1556  3              ident : $bvector [namb$s_vol_ident];
;  1473          1557  3          BIND
```

```
;  1474     1558  3          dev = namb_ptr [namb$q_device] : $desc_block,          ! Device string
;  1475     1559  3          dvi = namb_ptr [namb$q_device_dvi] : $desc_block;       ! Canonical name
;  1476     1560  3
;  1477     1561  3      ! If the device name parsed by RMS is known to us (has been MOUNTed), then the final vol_ident is
;  1478     1562  3      ! that device name.  A reference to a previously mounted virtual device will succeed here.
;  1479     1563  3      !
;  1480     1564  3      CH$COPY (.dev [dsc$w_length], .dev [dsc$a_pointer], 0, namb$s_vol_ident, ident);    ! Make a zero-padded
;  1481     1565  3      namb_ptr [namb$a_assoc_volb] = exch$util_find_mounted_volb (ident);                 ! Look at mounted vo
;  1482     1566  3      $trace_print_fao ('first try, ident "!AF", volb !XL', .dev [dsc$w_length], ident, .namb_ptr [namb$a_asso
;  1483     1567  3
;  1484     1568  3      ! If we have been given an address, then the ident is the correct name as it stands
;  1485     1569  3      !
;  1486     1570  3      IF .namb_ptr [namb$a_assoc_volb] NEQ 0
;  1487     1571  3      THEN
;  1488     1572  4          BEGIN
;  1489     1573  4
;  1490     1574  4          ! Copy the namb ident to the volume ident field
;  1491     1575  4          !
;  1492     1576  4          CH$MOVE (namb$s_vol_ident, ident, namb_ptr [namb$t_vol_ident]);
;  1493     1577  4          namb_ptr [namb$l_vol_ident_len] = .dev [dsc$w_length];
;  1494     1578  4          status = 1;
;  1495     1579  4
;  1496     1580  4          END
;  1497     1581  4
;  1498     1582  4      ! The device isn't mounted, did RMS find an unconcealed canonical device name for a non-virtual mount?
;  1499     1583  4      ! The virtual device parse option is only used when a virtual device is first mounted.  It is used to
;  1500     1584  4      ! prevent the virtual device "R:" from being interpreted as "_RTA0:", and "DM:" from turning into "_DMA0
;  1501     1585  4      ! On subsequent references to the virtual device we will get a match from the scan for mounted volumes,
;  1502     1586  4      ! and will not have to treat virtual device names as being special.
;  1503     1587  4      !
;  1504     1588  5      ELSE IF  (((.dvi [dsc$w_length] NEQ 0) AND (NOT .namb_ptr [namb$v_concealed_device]))
;  1505     1589  4            AND
;  1506     1590  4              (NOT .prsopt [prsopt$v_virtual_device]))
;  1507     1591  3      THEN
;  1508     1592  4          BEGIN
;  1509     1593  4          BIND
;  1510     1594  4              last = (.dvi [dsc$w_length] + namb_ptr [namb$t_vol_ident]) : BYTE;
;  1511     1595  4
;  1512     1596  4          ! Copy the canonical device name to the ident field, adding a colon
;  1513     1597  4          !
;  1514     1598  4          CH$MOVE (.dvi [dsc$w_length], .dvi [dsc$a_pointer], namb_ptr [namb$t_vol_ident]);
;  1515     1599  4          last = %C ':';
;  1516     1600  4          namb_ptr [namb$l_vol_ident_len] = .dvi [dsc$w_length] + 1;
;  1517     1601  4          status = 1;
;  1518     1602  4          $trace_print_fao ('canonical name, ident "!AF"', .namb_ptr [namb$l_vol_ident_len], namb_ptr [namb$t_
;  1519     1603  4
;  1520     1604  4          END
;  1521     1605  4
;  1522     1606  4      ! No canonical name, is there any device name at all?
;  1523     1607  4      !
;  1524     1608  3      ELSE IF .dev [dsc$w_length] NEQ 0
;  1525     1609  3          THEN
;  1526     1610  4          BEGIN
;  1527     1611  4
;  1528     1612  4          ! Copy the parsed device name to the ident field
;  1529     1613  4          !
;  1530     1614  4          CH$MOVE (.dev [dsc$w_length], .dev [dsc$a_pointer], namb_ptr [namb$t_vol_ident]);
```

```
;  1531          1615  4          namb_ptr [namb$l_vol_ident_len] = .dev [dsc$w_length];
;  1532          1616  4          status = 1;
;  1533          1617  4          $trace_print_fao ('parsed device name, ident "!AF"', .namb_ptr [namb$l_vol_ident_len], namb_ptr [nam
;  1534          1618  4
;  1535          1619  4          END
;  1536          1620  4
;  1537          1621  4      ! Wow - nothing at all
;  1538          1622  4      !
;  1539          1623  4      ELSE
;  1540          1624  4          status = 0;
;  1541          1625  2  END;
;  1542          1626  2
;  1543          1627  2  ! If we don't have a mounted volb address, then try again to see if our final ident is mounted
;  1544          1628  2  !
;  1545          1629  2  IF .namb_ptr [namb$a_assoc_volb] EQL 0
;  1546          1630  2  THEN
;  1547          1631  2      namb_ptr [namb$a_assoc_volb] = exch$util_find_mounted_volb (namb_ptr [namb$t_vol_ident]);
;  1548          1632  2  $trace_print_fao ('second check for volb, !XL', .namb_ptr [namb$a_assoc_volb]);
;  1549          1633  2
;  1550          1634  2  ! Let the user know if we are ignoring pieces of the file name
;  1551          1635  2  !
;  1552          1636  2  IF .namb_ptr [namb$a_assoc_volb] NEQ 0
;  1553          1637  2  THEN
;  1554          1638  3      BEGIN
;  1555          1639  3      BIND
;  1556          1640  3          volb = namb_ptr [namb$a_assoc_volb] : $ref_bblock;
;  1557          1641  3
;  1558          1642  3      ! RT-11 files have neither version numbers or directories
;  1559          1643  3      !
;  1560          1644  3      IF .volb [volb$b_vol_format] EQL volb$k_vfmt_rt11
;  1561          1645  3      THEN
;  1562          1646  4          BEGIN
;  1563          1647  4          IF .namb_ptr [namb$v_explicit_directory]
;  1564          1648  4          THEN
;  1565          1649  4              $exch_signal (exch$_ignore_dire);
;  1566          1650  4          IF .namb_ptr [namb$v_explicit_version]
;  1567          1651  4          THEN
;  1568          1652  4              $exch_signal (exch$_ignore_vers);
;  1569          1653  3          END;
;  1570          1654  3
;  1571          1655  3      ! DOS-11 files do not have version numbers
;  1572          1656  3      !
;  1573          1657  3      IF .volb [volb$b_vol_format] EQL volb$k_vfmt_dos11
;  1574          1658  3      THEN
;  1575          1659  4          BEGIN
;  1576          1660  4          IF .namb_ptr [namb$v_explicit_version]
;  1577          1661  4          THEN
;  157?          1662  4              $exch_signal (exch$_ignore_vers);
;  1579          1663  3          END;
;  1580          1664  2      END;
;  1581          1665  2
;  1582          1666  2  ! If we have more files then set the more files flag
;  1583          1667  2  !
;  1584          1668  2  IF .cli_status EQL cli$_comma                          ! Separated by a comma
;  1585          1669  2   OR
;  1586          1670  2      .cli_status EQL cli$_concat                         ! Or by a plus sign
;  1587          1671  2  THEN
```

```
;  1588          1672  2      namb_ptr [namb$v_more_files] = true;
;  1589          1673  2
;  1590        P 1674  2 $trace_print_fao ('id "!AF" volb !XL parse !XL "!AS" "!AS" "!AS" "!AS" "!AS"  "!AS"',
;  1591        P 1675  2          .namb_ptr [namb$l_vol_ident_len], namb_ptr [namb$t_vol_ident],
;  1592        P 1676  2          .namb_ptr [namb$a_assoc_volb], .prs_status,
;  1593        P 1677  2          namb_ptr [namb$q_node],  namb_ptr [namb$q_device],  namb_ptr [namb$q_directory],
;  1594          1678  2          namb_ptr [namb$q_name],  namb_ptr [namb$q_type],    namb_ptr [namb$q_version]);
;  1595          1679  2
;  1596          1680  2 ! Return the pointer and the result status
;  1597          1681  2 !
;  1598          1682  2 .namb = .namb_ptr;
;  1599          1683  2 RETURN .status;
;  1600          1684  1 END;
;  INFO#250                L1:1481
;  Referenced LOCAL symbol DEVNAMLEN is probably not initialized
;  INFO#250                L1:1529
;  Referenced LOCAL symbol DEVCLASS is probably not initialized
;  INFO#250                L1:1530
;  Referenced LOCAL symbol DEVTYPE is probably not initialized


                                                        .EXTRN  SYS$PARSE, SYS$GETDVIW
                                                        .EXTRN  EXCH$_ACCESSFAIL
                                                        .EXTRN  EXCH$_IGNORE_DIRE
                                                        .EXTRN  EXCH$_IGNORE_VERS
                                                        .EXTRN  CLI$_COMMA, CLI$_CONCAT

                                        0FFC 00000      .ENTRY  EXCH$CMD_PARSE_FILESPEC, Save R2,R3,R4,R5,- ; 1330
                                                                R6,R7,R8,R9,R10,R11
                                    5E  FD80   CE  9E 00002     MOVAB   -640(SP), SP
                                        14     BC  D4 00007     CLRL    @NAMB                               ; 1393
                                    56  10     AC  D0 0000A     MOVL    NAME_DESC, R6                       ; 1400
                                0E      0C  AC     01  E1 0000E BBC     #1, PRSOPT, 1$                      ; 1398
                                        04  AC     DD  00013    PUSHL   PARA_NAME                           ; 1400
                                            56     DD  00016    PUSHL   R6
                            00000000G   00 00      02  FB 00018 CALLS   #2, STR$COPY_DX
                                            15     11  0001F    BRB     2$
                                            56     DD  00021 1$:PUSHL   R6                                 ; 1403
                                        04  AC     DD  00023    PUSHL   PARA_NAME
                            00000000G   00 00      02  FB 00026 CALLS   #2, CLI$GET_VALUE
                                        6E 50      D0  0002D    MOVL    R0, CLI_STATUS
                                        03 6E      E8  00030    BLBS    CLI_STATUS, 2$
                                               02B3 31  00033    BRW     28$
0050    8F      00                      6E         00  2C 00036 2$:MOVC5   #0, (SP), #0, #80, $RMS_PTR      ; 1414
                                                B0 AD  0003D
                                    B0  AD  5003   8F  B0 0003F    MOVW    #20483, $RMS_PTR
                                    C6  AD         02  90 00045    MOVB    #2, $RMS_PTR+22
                                    CF  AD         02  90 00049    MOVB    #2, $RMS_PTR+31
                                    D8  AD  FF50   CD  9E 0004D    MOVAB   NAM, $RMS_PTR+40
                                    DC  AD  04     A6  D0 00053    MOVL    4(R6), $RMS_PTR+44
                                    E4  AD         66  90 00058    MOVB    (R6), $RMS_PTR+52
0060    8F      00                      6E         00  2C 0005C    MOVC5   #0, (SP), #0, #96, $RMS_PTR      ; 1419
                                                FF50 CD  00063
                                FF50    CD  6002   8F  B0 00066    MOVW    #24578, $RMS_PTR
                                FF5A    CD         01  8E 0006D    MNEGB   #1, $RMS_PTR+10
                                FF5C    CD 00D0    CE  9E 00072    MOVAB   EBUF, $RMS_PTR+12
```

```
                    50      08   AC  D0 00079          MOVL    DEFAULT_DESC, R0                        ; 1424
                                 09  13 0007D          BEQL    3$
                 E5 AD           60  90 0007F          MOVB    (R0), FAB+53                            ; 1427
                 E0 AD       04  A0  D0 00083          MOVL    4(R0), FAB+48                           ; 1428
                             B0  AD  9F 00088 3$:      PUSHAB  FAB                                     ; 1435
          00000000G  00          01  FB 0008B          CALLS   #1, SYS$PARSE
                     52          50  D0 00092          MOVL    R0, PRS_STATUS
                             B0  AD  9F 00095          PUSHAB  FAB                                     ; 1436
               0000V CF          01  FB 00098          CALLS   #1, EXCH$CMD_PARSE_NULL_FILE
                         FF5B    CD  95 0009D          TSTB    NAM+11                                  ; 1444
                                 04  12 000A1          BNEQ    4$
                     50          52  D0 000A3          MOVL    PRS_STATUS, R0                          ; 1446
                                 04  000A6             RET
          00018272   8F          52  D1 000A7 4$:      CMPL    PRS_STATUS, #98930                      ; 1453
                                 03  13 000AE          BEQL    5$
                             00A5 31  000B0             BRW     9$
                         89      AD  95 000B3 5$:      TSTB    NAM+57                                  ; 1459
                                 13  12 000B6          BNEQ    6$
                     7E      68  8F  9A 000B8          MOVZBL  #104, -(SP)
                                 01  DD 000BC          PUSHL   #1
                         00000000G   8F  DD 000BE      PUSHL   #EXCH$_BADLOGIC
          00000000G  00          03  FB 000C4          CALLS   #3, LIB$STOP
               00A2  CE      010E  8F  B0 000CB 6$:    MOVW    #270, DESC+2                            ; 1460
               00A0  CE        89  AD  9B 000D2        MOVZBW  NAM+57, DESC
               00A4  CE        94  AD  D0 000D8        MOVL    NAM+68, DESC+4
               00A8  CE  00E80010  8F  D0 000DE        MOVL    #15204368, DEV_ITEM                     ; 1465
               00AC  CE      0090  CE  9E 000E7        MOVAB   DEVNAM, DEV_ITEM+4                      ; 1466
               00B0  CE        04  AE  9E 000EE        MOVAB   DEVNAMLEN, DEV_ITEM+8                   ; 1467
               00B4  CE  00020004  8F  D0 000F4        MOVL    #131076, DEV_ITEM+12                    ; 1468
               00B8  CE        F0  AD  9E 000FD        MOVAB   FAB+64, DEV_ITEM+16                     ; 1469
                             00BC  CE  7C 00103        CLRQ    DEV_ITEM+20                             ; 1470
                                 7E  7C 00107          CLRQ    -(SP)                                   ; 1475
                                 7E  7C 00109          CLRQ    -(SP)
                             00B8  CE  9F 0010B        PUSHAB  DEV_ITEM
                             00B4  CE  9F 0010F        PUSHAB  DEV_DESC
                                 7E  7C 00113          CLRQ    -(SP)
                00000000G  00          08  FB 00115          CALLS   #8, SYS$GETDVIW
                           5B          50  D0 0011C          MOVL    R0, STATUS
                           09          5B  E8 0011F          BLBS    STATUS, 7$
                                       5B  DD 00122          PUSHL   STATUS                          ; 1477
                                   00A4 CE  9F 00124         PUSHAB  DEV_DESC
                                   009E 31  00128            BRW     10$
                           0F      04  AE  B1 0012B 7$:     CMPW    DEVNAMLEN, #15                  ; 1481
                                       13  1B 0012F         BLEQU   8$
                           7E      69  8F  9A 00131         MOVZBL  #105, -(SP)
                                       01  DD 00135         PUSHL   #1
                                00000000G  8F  DD 00137     PUSHL   #EXCH$_BADLOGIC
                00000000G  00          03  FB 0013D          CALLS   #3, LIB$STOP
          FF64  CD      04  AE          01  83 00144 8$:     SUBB3   #1, DEVNAMLEN, NAM_DVILEN       ; 1482
                           50      FF64  CD  9A 00148        MOVZBL  NAM_DVILEN, R0                  ; 1483
          FF65  CD    0090  CE          50  28 00150        MOVC3   R0, DEVNAM, NAM_DVIBUF
                00000000G  EF          00  FB 00158 9$:     CALLS   #0, EXCH$UTIL_NAMB_ALLOCATE     ; 1490
                           56          50  D0 0015F          MOVL    R0, NAMB_PTR
                                       56  DD 00162          PUSHL   NAMB_PTR                        ; 1494
                                   B0  AD  9F 00164          PUSHAB  FAB
                             FBE6  CF      02  FB 00167      CALLS   #2, CMD_NAMB_FAB_COPY
                           5A      30  A6  9E 0016C          MOVAB   48(NAMB_PTR), R10               ; 1501
```

```
                                        6A  B5 00170          TSTW    (R10)                                   ; 1503
                                        6F  13 00172          BEQL    12$                                     ;
                          00A8  CE 00040004 8F  D0 00174          MOVL    #262148, DEV_ITEM                       ; 1513
                          00AC  CE       08 AE  9E 0017D          MOVAB   DEVCLASS, DEV_ITEM+4                    ; 1514
                                      00B0  CE  D4 00183          CLRL    DEV_ITEM+8                              ; 1515
                          00B4  CE 00060004 8F  D0 00187          MOVL    #393220, DEV_ITEM+12                    ; 1516
                          00B8  CE       0C AE  9E 00190          MOVAB   DEVTYPE, DEV_ITEM+16                    ; 1517
                                      00BC  CE  D4 00196          CLRL    DEV_ITEM+20                             ; 1518
                          00C0  CE 00020004 8F  D0 0019A          MOVL    #131076, DEV_ITEM+24                    ; 1519
                          00C4  CE       68 A6  9E 001A3          MOVAB   104(R6), DEV_ITEM+28                    ; 1520
                                      00C8  CE  7C 001A9          CLRQ    DEV_ITEM+32                             ; 1521
                                            7E  7C 001AD          CLRQ    -(SP)                                   ; 1526
                                            7E  7C 001AF          CLRQ    -(SP)                                   ;
                                      00B8  CE  9F 001B1          PUSHAB  DEV_ITEM                                ;
                                            5A  DD 001B5          PUSHL   R10                                     ;
                                            7E  7C 001B7          CLRQ    -(SP)                                   ;
                00000000G   00              08  FB 001B9          CALLS   #8, SYS$GETDVIW                         ;
                            5B              50  D0 001C0          MOVL    R0, STATUS                              ;
                            13              5B  E8 001C3          BLBS    STATUS, 11$                             ;
                            7E              5A  7D 001C6          MOVQ    R10, -(SP)                              ; 1528
                                            01  DD 001C9 10$:     PUSHL   #1                                      ;
                                 00000000G  8F  DD 001CB          PUSHL   #EXCH$_ACCESSFAIL                       ;
                00000000G   00              04  FB 001D1          CALLS   #4, LIB$STOP                            ;
                                                04 001D8          RET                                             ;
                          78  A6        08  AE  90 001D9 11$:     MOVB    DEVCLASS, 120(NAMB_PTR)                 ; 1529
                          79  A6        0C  AE  90 001DE          MOVB    DEVTYPE, 121(NAMB_PTR)                  ; 1530
                                            56  DD 001E3 12$:     PUSHL   NAMB_PTR                                ; 1538
                        F5C3  CF              01  FB 001E5          CALLS   #1, EXCH$CMD_FETCH_REC_FORMAT           ;
                            5B              50  D0 001EA          MOVL    R0, STATUS                              ;
                            0D              5B  E9 001ED          BLBC    STATUS, 13$                             ;
                                            56  DD 001F0          PUSHL   NAMB_PTR                                ; 1545
                                  F8E9      CF  01  FB 001F2          CALLS   #1, EXCH$CMD_FETCH_VOL_FORMAT           ;
                                            5B  50  D0 001F7          MOVL    R0, STATUS                              ;
                                            0C  5B  E8 001FA          BLBS    STATUS, 14$                             ;
                                            56  DD 001FD 13$:     PUSHL   NAMB_PTR                                ; 1548
                                 00000000G  EF  01  FB 001FF          CALLS   #1, EXCH$UTIL_NAMB_RELEASE              ;
                                                00DC  31 00206          BRW     27$                                     ; 1549
                                            57  40  A6  9E 00209 14$:     MOVAB   64(NAMB_PTR), R7                        ; 1558
0080  8F                  00        04  B7  67  2C 0020D          MOVC5   (R7), @4(R7), #0, #128, IDENT           ; 1564
                                                10  AE  00215                                                  ;
                                            59  74  A6  9E 00217          MOVAB   116(NAMB_PTR), R9                       ; 1565
                                                10  AE  9F 0021B          PUSHAB  IDENT                                   ;
                                 00000000G  EF  01  FB 0021E          CALLS   #1, EXCH$UTIL_FIND_MOUNTED_VOLB         ;
                                            69  50  D0 00225          MOVL    R0, (R9)                                ;
                                                0B  13 00228          BEQL    15$                                     ; 1570
                      008A  C6        10  AE  0080  8F  28 0022A          MOVC3   #128, IDENT, 138(NAMB_PTR)              ; 1576
                                                31  11 00233          BRB     17$                                     ; 1577
                                            58  6A  3C 00235 15$:     MOVZWL  (R10), R8                               ; 1588
                                                21  13 00238          BEQL    16$                                     ;
                            1C        6D  A6  04  E0 0023A          BBS     #4, 109(NAMB_PTR), 16$                  ;
                                                18  0C  AC  E8 0023F          BLBS    PRSOPT, 16$                             ; 1590
                                                50  56  D0 00243          MOVL    NAMB_PTR, R0                            ; 1598
                      008A  C0        04  BA  58  28 00246          MOVC3   R8, @4(R10), 138(R0)                    ;
                                  008A  C648  3A  90 0024D          MOVB    #58, 138(NAMB_PTR)[R8]                  ; 1599
                                  0086  C6    01  A8  9E 00253          MOVAB   1(R8), 134(NAMB_PTR)                    ; 1600
                                                10  11 00259          BRB     18$                                     ; 1601
                                                67  B5 0025B 16$:     TSTW    (R7)                                    ; 1608
```

```
                                                11  13  0025D                BEQL    19$                                                     ;
                                008A    C6      67  28  0025F                MOVC3   (R7), @4(R7), 138(NAMB_PTR)                           ; 1614
                                0086    C6      67  3C  00266 17$:           MOVZWL  (R7), 134(NAMB_PTR)                                     ; 1615
                                        5B      01  D0  0026B 18$:           MOVL    #1, STATUS                                              ; 1616
                                                02  11  0026E                BRB     20$                                                     ; 1608
                                                5B  D4  00270 19$:           CLRL    STATUS                                                  ; 1624
                                                69  D5  00272 20$:           TSTL    (R9)                                                    ; 1629
                                                10  12  00274                BNEQ    21$                                                     ;
                                        008A    C6  9F  00276                PUSHAB  138(NAMB_PTR)                                           ; 1631
                            00000000G   EF          01  FB  0027A            CALLS   #1, EXCH$UTIL_FIND_MOUNTED_VOLB                         ;
                                        69          50  D0  00281            MOVL    R0, (R9)                                                ;
                                                44  13  00284                BEQL    24$                                                     ;
                                        52          69  D0  00286 21$:       MOVL    (R9), R2                                                ; 1636
                                        03      58  A2  91  00289            CMPB    88(R2), #3                                              ; 1644
                                                23  12  0028D                BNEQ    23$                                                     ;
                                        0D      6D  A6  E9  0028F            BLBC    109(NAMB_PTR), 22$                                      ;
                                            00000000G   8F  DD  00293        PUSHL   #EXCH$_IGNORE_DIRE                                      ; 1647
                            00000000G   00          01  FB  00299            CALLS   #1, LIB$SIGNAL                                          ; 1649
                        0D      6D      A6  03  E1  002A0 22$:               BBC     #3, 109(NAMB_PTR), 23$                                  ;
                                            00000000G   8F  DD  002A5        PUSHL   #EXCH$_IGNORE_VERS                                      ; 1650
                            00000000G   00          01  FB  002AB            CALLS   #1, LIB$SIGNAL                                          ; 1652
                                        01      58  A2  91  002B2 23$:       CMPB    88(R2), #1                                              ;
                                                12  12  002B6                BNEQ    24$                                                     ; 1657
                        0D      6D      A6  03  E1  002B8                    BBC     #3, 109(NAMB_PTR), 24$                                  ;
                                            00000000G   8F  DD  002BD        PUSHL   #EXCH$_IGNORE_VERS                                      ; 1660
                            00000000G   00          01  FB  002C3            CALLS   #1, LIB$SIGNAL                                          ; 1662
                            00000000G   8F      6E  D1  002CA 24$:           CMPL    CLI_STATUS, #CLI$_COMMA                                 ;
                                                09  13  002D1                BEQL    25$                                                     ; 1668
                            00000000G   8F      6E  D1  002D3                CMPL    CLI_STATUS, #CLI$_CONCAT                                ;
                                                05  12  002DA                BNEQ    26$                                                     ; 1670
                                6D      A6      80  8F  88  002DC 25$:       BISB2   #128, 109(NAMB_PTR)                                     ; 1672
                                14      BC          56  D0  002E1 26$:       MOVL    NAMB_PTR, @NAMB                                         ; 1682
                                        50          5B  D0  002E5 27$:       MOVL    STATOS, R0                                              ; 1683
                                                        04  002E8             RET                                                             ;
                                                    50  D4  002E9 28$:       CLRL    R0                                                      ; 1684
                                                        04  002EB             RET                                                             ;

; Routine Size:  748 bytes,    Routine Base:  EXCH$CMD_CODE + 0A16
```

```
;  1602     1685  1 GLOBAL ROUTINE exch$cmd_parse_null_file (infab : $ref_bblock) : NOVALUE =       %SBTTL 'exch$cmd_parse_null_
;  1603     1686  2 BEGIN
;  1604     1687  2 !++
;  1605     1688  2 !
;  1606     1689  2 ! FUNCTIONAL DESCRIPTION:
;  1607     1690  2 !
;  1608     1691  2 !       This routine makes a copy of the input fab (and nam block, if present).  All name fields are set to
;  1609     1692  2 !       zero, and a $parse is done.  This causes RMS to deassign all context for the fab.
;  1610     1693  2 !
;  1611     1694  2 ! INPUTS:
;  1612     1695  2 !
;  1613     1696  2 !       infab - address of fab
;  1614     1697  2 !
;  1615     1698  2 ! IMPLICIT INPUTS:
;  1616     1699  2 !
;  1617     1700  2 !       none
;  1618     1701  2 !
;  1619     1702  2 ! OUTPUTS:
;  1620     1703  2 !
;  1621     1704  2 !       none
;  1622     1705  2 !
;  1623     1706  2 ! IMPLICIT OUTPUTS:
;  1624     1707  2 !
;  1625     1708  2 !       none
;  1626     1709  2 !
;  1627     1710  2 ! ROUTINE VALUE:
;  1628     1711  2 !
;  1629     1712  2 !       none
;  1630     1713  2 !
;  1631     1714  2 ! SIDE EFFECTS:
;  1632     1715  2 !
;  1633     1716  2 !       RMS internal context released
;  1634     1717  2 !--
;  1635     1718  2
;  1636     1719  2 $dbgtrc_prefix ('cmd_parse_null_file> ');
;  1637     1720  2
;  1638     1721  2 LOCAL
;  1639     1722  2     fab         : $bblock [fab$k_bln],          ! FAB for the RMS parse
;  1640     1723  2     nam         : $bblock [nam$k_bln]           ! NAM for the file name
;  1641     1724  2     ;
;  1642     1725  2
;  1643     1726  2 !
;  1644     1727  2 ! Copy the input fab to the local and zero the file name lengths
;  1645     1728  2 !
;  1646     1729  2 CH$MOVE (fab$k_bln, .infab, fab);                ! Copy to local
;  1647     1730  2 fab [fab$b_fns] = 0;                             ! File name size to zero
;  1648     1731  2 fab [fab$b_dns] = 0;                             ! Default name size to zero
;  1649     1732  2
;  1650     1733  2 !
;  1651     1734  2 ! If a nam block is present on the input, set up a local nam block
;  1652     1735  2 !
;  1653     1736  2 IF .fab [fab$l_nam] NEQ 0
;  1654     1737  2 THEN
;  1655     1738  3     BEGIN
;  1656     1739  3     CH$MOVE (nam$k_bln, .fab [fab$l_nam], nam); ! Copy the old nam block
;  1657     1740  3     nam [nam$v_svctx] = 0;                      ! Clear the save context bit
;  1658     1741  3     nam [nam$v_synchk] = 1;                     ! Syntax check only
```

```
;  1659     1742  3      nam [nam$b_esl] = 0;                    ! No expanded string
;  1660     1743  3      nam [nam$b_ess] = 0;                    ! No expanded string buffer
;  1661     1744  3      nam [nam$b_rsl] = 0;                    ! No result string
;  1662     1745  3      nam [nam$b_rss] = 0;                    ! No result string buffer
;  1663     1746  3      nam [nam$l_rlf] = 0;                    ! No related file nam
;  1664     1747  3      fab [fab$l_nam] = nam;                  ! Point the fab at this one
;  1665     1748  2      END;
;  1666     1749  2
;  1667     1750  2  !
;  1668     1751  2  ! Now do the dummy parse
;  1669     1752  2  !
;  1670     1753  2  $parse (FAB = fab);                         ! Parse the null name
;  1671     1754  2
;  1672     1755  2  RETURN;
;  1673     1756  1  END;


                                                 003C 00000     .ENTRY  EXCH$CMD_PARSE_NULL_FILE, Save R2,R3,R4,R5  ; 1685
                                      5E   FF50  CE   9E 00002  MOVAB   -176(SP), SP                                ;
                 60   AE       04    BC   0050  8F   28 00007  MOVC3   #80, @INFAB, FAB                            ; 1729
                                           E4   AD   B4 0000F  CLRW    FAB+52                                      ; 1730
                                           D8   AD   D5 00012  TSTL    FAB+40                                      ; 1736
                                                1D   13 00015  BEQL    1$                                          ;
                      6E       D8   BD   0060  8F   28 00017  MOVC3   #96, @FAB+40, NAM                          ; 1739
                               33   AE     80  8F   8A 0001E  BICB2   #128, NAM+51                                ; 1740
                               08   AE         08   88 00023  BISB2   #8, NAM+8                                   ; 1741
                                         0A   AE   B4 00027  CLRW    NAM+10                                      ; 1743
                                         02   AE   B4 0002A  CLRW    NAM+2                                       ; 1745
                                         10   AE   D4 0002D  CLRL    NAM+16                                      ; 1746
                                    D8   AD        6E   9E 00030  MOVAB   NAM, FAB+40                            ; 1747
                                         60   AE   9F 00034  1$:    PUSHAB  FAB                                     ; 1753
                           00000000G     00        01   FB 00037  CALLS   #1, SYS$PARSE                          ;
                                                    04   0003E  RET                                               ; 1756

; Routine Size:  63 bytes,    Routine Base:  EXCH$CMD_CODE + 0D02
```

```
;  1675          1757  1 GLOBAL ROUTINE exch$cmd_related_file_fixup (nam : $ref_bblock) : NOVALUE =      %SBTTL 'exch$cmd_related_fil
;  1676          1758  2 BEGIN
;  1677          1759  2 !++
;  1678          1760  2 !
;  1679          1761  2 ! FUNCTIONAL DESCRIPTION:
;  1680          1762  2 !
;  1681          1763  2 !       This routine is called when exch$cmd_related_file_parse finds that the related file name is a quoted
;  1682          1764  2 !       file spec (NAM$V_QUOTED set).  It attempts to determine if the filename is from an RT-11 magtape, in
;  1683          1765  2 !       which case it converts the quoted file spec into a normal file spec by removing the quotes and embed
;  1684          1766  2 !       spaces.  If the quoted name is not an RT-11 filespec, no action is taken.
;  1685          1767  2 !
;  1686          1768  2 ! INPUTS:
;  1687          1769  2 !
;  1688          1770  2 !       nam - address of an RMS NAM block containing the quoted file name in the expanded string buffer
;  1689          1771  2 !
;  1690          1772  2 ! IMPLICIT INPUTS:
;  1691          1773  2 !
;  1692          1774  2 !       none
;  1693          1775  2 !
;  1694          1776  2 ! OUTPUTS:
;  1695          1777  2 !
;  1696          1778  2 !       nam - expanded string is modified in place, NAM$B_ESL is adjusted for the new length.  Other nam fie
;  1697          1779  2 !             are not corrected
;  1698          1780  2 !
;  1699          1781  2 ! IMPLICIT OUTPUTS:
;  1700          1782  2 !
;  1701          1783  2 !       none
;  1702          1784  2 !
;  1703          1785  2 ! ROUTINE VALUE:
;  1704          1786  2 !
;  1705          1787  2 !       none
;  1706          1788  2 !
;  1707          1789  2 ! SIDE EFFECTS:
;  1708          1790  2 !
;  1709          1791  2 !       none
;  1710          1792  2 !--
;  1711          1793  2
;  1712          1794  2 $dbgtrc_prefix ('cmd_related_file_fixup> ');
;  1713          1795  2
;  1714          1796  2 LOCAL
;  1715          1797  2     il,
;  1716          1798  2     skipped,                                        ! Count of characters skipped
;  1717          1799  2     ip : $ref_bvector,                              ! Input pointer to name
;  1718          1800  2     op : $ref_bvector,
;  1719          1801  2     dots
;  1720          1802  2     ;
;  1721          1803  2
;  1722        P 1804  2 $trace_print_fao ('expanded "!AF", quoted name {!AF}, type {!AF}, ver {!AF}',
;  1723        P 1805  2                   .nam [nam$b_esl], .nam [nam$l_esa], .nam [nam$b_name], .nam [nam$l_name],
;  1724          1806  2                   .nam [nam$b_type], .nam [nam$l_type], .nam [nam$b_ver], .nam [nam$l_ver]);
;  1725          1807  2
;  1726          1808  2 ! Get length and address of the file NAME, without the quotes
;  1727          1809  2 !
;  1728          1810  2 il = .nam [nam$b_name] - 2;
;  1729          1811  2 IF .il LEQ 0
;  1730          1812  2 THEN
;  1731          1813  2     RETURN;                                         ! Can't be RT-11 if it is zero length
```

```
;  1732          1814  2 IF .nam [nam$b_type] NEQ 1                        ! Type field should also be length of one
;  1733          1815  2 THEN
;  1734          1816  2     RETURN;
;  1735          1817  2 op = .nam [nam$l_name];
;  1736          1818  2 ip = .op + 1;
;  1737          1819  2
;  1738          1820  2 ! Check that the name contains the proper set of characters
;  1739          1821  2 !
;  1740          1822  2 dots = 0;                                          ! Assume no periods in the name
;  1741          1823  2 INCR p FROM 0 TO .il-1
;  1742          1824  2 DO
;  1743          1825  2     SELECTONE .ip [.p] OF
;  1744          1826  2     SET
;  1745          1827  2         ['A' TO 'Z', '0' TO '9', ' '] : ;                   ! Alphanumerics and spaces are ok
;  1746          1828  2         ['.'] :                         dots = .dots + 1;   ! Count the period
;  1747          1829  2         [OTHERWISE] :                   RETURN;             ! Can't be RT-11 file name
;  1748          1830  2     TES;
;  1749          1831  2
;  1750          1832  2 ! We can only have zero or one periods in the name
;  1751          1833  2 !
;  1752          1834  2 IF .dots GTRU 1
;  1753          1835  2 THEN
;  1754          1836  2     RETURN;
;  1755          1837  2
;  1756          1838  2 ! Shuffle the name, skipping any spaces
;  1757          1839  2 !
;  1758          1840  2 skipped = 3;                                       ! Init to the two quotes being skipped plus type
;  1759          1841  2 INCR p FROM 0 to .il-1
;  1760          1842  2 DO
;  1761          1843  3     BEGIN
;  1762          1844  3     REGISTER
;  1763          1845  3         char;
;  1764          1846  3     char = CH$RCHAR_A (ip);                       ! Grab the character
;  1765          1847  3     IF .char EQL ' '
;  1766          1848  3     THEN
;  1767          1849  3         skipped = .skipped + 1                    ! Bump count if space
;  1768          1850  3     ELSE
;  1769          1851  3         CH$WCHAR_A (.char, op);                   ! Otherwise copy to output
;  1770          1852  2     END;
;  1771          1853  2 nam [nam$b_esl] = .nam [nam$b_esl] - .skipped;    ! Store the new length
;  1772          1854  2
;  1773          1855  2 ! Now slide the remaining field (VERSION) over
;  1774          1856  2 !
;  1775          1857  2 CH$MOVE (.nam [nam$b_ver], .nam [nam$l_ver], .op);
;  1776          1858  2
;  1777          1859  2 $trace_print_fao ('modified name "!AF"', .nam [nam$b_esl], .nam [nam$l_esa]);
;  1778          1860  2
;  1779          1861  2 RETURN;
;  1780          1862  1 END;


                                        007C 00000          .ENTRY  EXCH$CMD_RELATED_FILE_FIXUP, Save R2,R3,R4,-; 1757
                                                                    R5,R6
                                53      04    AC  D0 00002          MOVL    NAM, R3                                ; 1810
```

```
                                56        3B    A3  9A 00006          MOVZBL  59(R3), IL                      ;
                                56              02  C2 0000A          SUBL2   #2, IL                          ;
                                                6C  15 0000D          BLEQ    8$                              ; 1811
                                01        3C    A3  91 0000F          CMPB    60(R3), #1                      ; 1814
                                                66  12 00013          BNEQ    8$
                                52        4C    A3  D0 00015          MOVL    76(R3), OP                      ; 1817
                                51        01    A2  9E 00019          MOVAB   1(R2), IP                       ; 1818
                                                55  D4 0001D          CLRL    DOTS                            ; 1822
                                54              01  CE 0001F          MNEGL   #1, P                           ; 1825
                                                26  11 00022          BRB     4$
                                50            6441  9A 00024 1$:      MOVZBL  (P)[IP], R0                     ;
                                20              50  91 00028          CMPB    R0, #32                         ; 1827
                                                1D  13 0002B          BEQL    4$
                                30              50  91 0002D          CMPB    R0, #48                         ;
                                                05  1F 00030          BLSSU   2$                              ;
                                39              50  91 00032          CMPB    R0, #57                         ;
                                                13  1B 00035          BLEQU   4$                              ;
                             41 8F              50  91 00037 2$:      CMPB    R0, #65                         ;
                                                06  1F 0003B          BLSSU   3$                              ;
                             5A 8F              50  91 0003D          CMPB    R0, #90                         ;
                                                07  1B 00041          BLEQU   4$                              ;
                                2E              50  91 00043 3$:      CMPB    R0, #46                         ; 1828
                                                33  12 00046          BNEQ    8$                              ;
                                                55  D6 00048          INCL    DOTS                            ;
                          D6                    56  F2 0004A 4$:      AOBLSS  IL, P, 1$                       ; 1825
                                01              55  D1 0004E          CMPL    DOTS, #1                        ; 1834
                                                28  1A 00051          BGTRU   8$                              ;
                                55              03  D0 00053          MOVL    #3, SKIPPED                     ; 1840
                                54              01  CE 00056          MNEGL   #1, P                           ; 1841
                                                0F  11 00059          BRB     7$                              ;
                                50              81  9A 0005B 5$:      MOVZBL  (IP)+, CHAR                     ; 1846
                                20              50  D1 0005E          CMPL    CHAR, #32                       ; 1847
                                                04  12 00061          BNEQ    6$                              ;
                                                55  D6 00063          INCL    SKIPPED                         ; 1849
                                                03  11 00065          BRB     7$                              ; 1851
                                82              50  90 00067 6$:      MOVB    CHAR, (OP)+                     ;
                          ED    54              56  F2 0006A 7$:      AOBLSS  IL, P, 5$                       ; 1841
                             0B A3              55  82 0006E          SUBB2   SKIPPED, 11(R3)                 ; 1853
                                50        3D    A3  9A 00072          MOVZBL  61(R3), R0                      ; 1857
                          62    54 83              50  28 00076          MOVC3   R0, @84(R3), (OP)               ;
                                                    04  0007B 8$:      RET                                     ; 1862

; Routine Size:  124 bytes,    Routine Base:  EXCH$CMD_CODE + 0D41
```

```
;  1782          1863  1 GLOBAL ROUTINE exch$cmd_related_file_parse (fil_len, fil_buf : $ref_bvector,    %SBTTL 'exch$cmd_related_fil
;  1783          1864  1                                 rlf_len, rlf_buf : $ref_bvector, nam : $ref_bblock) =
;  1784          1865  2 BEGIN
;  1785          1866  2 !++
;  1786          1867  2 !
;  1787          1868  2 ! FUNCTIONAL DESCRIPTION:
;  1788          1869  2 !
;  1789          1870  2 !       This routine produces a file name for an output file by doing an RMS file parse to combine the
;  1790          1871  2 !       requested name (fil_len:fil_buf) with the related name (rlf_len:rlf_buf).  The output is placed in
;  1791          1872  2 !       a block which is an RMS NAM block (size NAM$C_BLN) immediately followed by the expanded string buffe
;  1792          1873  2 !       (size NAM$C_MAXRSS).
;  1793          1874  2 !
;  1794          1875  2 ! INPUTS:
;  1795          1876  2 !
;  1796          1877  2 !       fil_len - length of the requested name, possibly containing wildcards
;  1797          1878  2 !       fil_buf - address of
;  1798          1879  2 !       rlf_len - length of the related file name
;  1799          1880  2 !       rlf_buf - address of
;  1800          1881  2 !
;  1801          1882  2 ! IMPLICIT INPUTS:
;  1802          1883  2 !
;  1803          1884  2 !       none
;  1804          1885  2 !
;  1805          1886  2 ! OUTPUTS:
;  1806          1887  2 !
;  1807          1888  2 !       nam - address of an RMS NAM block and expanded string buffer
;  1808          1889  2 !
;  1809          1890  2 ! IMPLICIT OUTPUTS:
;  1810          1891  2 !
;  1811          1892  2 !       none
;  1812          1893  2 !
;  1813          1894  2 ! ROUTINE VALUE:
;  1814          1895  2 !
;  1815          1896  2 !       True if success, bad status code if unable to perform the operation
;  1816          1897  2 !
;  1817          1898  2 ! SIDE EFFECTS:
;  1818          1899  2 !
;  1819          1900  2 !       none
;  1820          1901  2 !--
;  1821          1902  2
;  1822          1903  2 $dbgtrc_prefix ('cmd_related_file_parse> ');
;  1823          1904  2
;  1824          1905  2 LOCAL
;  1825          1906  2     fab         : $bblock [fab$k_bln],          ! FAB for the RMS parse
;  1826          1907  2     rlf_nam     : $bblock [nam$k_bln],          ! NAM for the related file name
;  1827          1908  2     rbuf        : $bblock [nam$c_maxrss],       ! Buffer for the related file name
;  1828          1909  2     status                                      ! Status from the $parse
;  1829          1910  2     ;
;  1830          1911  2
;  1831          1912  2 $trace_print_fao ('input name "!AF", related name "!AF"', .fil_len, .fil_buf, .rlf_len, .rlf_buf);
;  1832          1913  2
;  1833          1914  2 ! Perform a $PARSE to create an RMS nam block for the related file name
;  1834          1915  2 !
;  1835        P 1916  2 $fab_init (
;  1836        P 1917  2           fab = fab,                            ! Input file FAB
;  1837        P 1918  2           fna = .rlf_buf,                       !  Set related name addr
;  1838        P 1919  2           fns = .rlf_len,                       !  Set related name size
```

```
                nam = rlf_nam);                          ! Name block
        $nam_init (
                nam = rlf_nam,                           ! File name block
                esa = rbuf,                              !  Result name buffer addr
                ess = nam$c_maxrss,                      !  Result name buffer size
                nop = synchk);                           !  Syntax check only

        ! Parse the input related file name into the nam block
        !
        $debug_print_lit ('fab before related parse (input)   <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<');
        $check_call (5, exch$dbg_fab_dump, fab);         ! Dump the fab
        status = $parse (FAB = fab);                     ! Parse the related name
        $debug_print_lit ('fab after related parse (input)   >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>');
        $check_call (5, exch$dbg_fab_dump, fab);         ! Dump the fab
        IF .rlf_nam [nam$b_esl] EQL 0                    ! It shouldn't be possible to get this serious an error sinc
        THEN                                             !  related file name has already been through a $parse
            $exch_signal_stop (.status);

        ! If the related name is a quoted string, see if it could be a quoted RT-11 magtape file name
        !
        IF .rlf_nam [nam$v_quoted]
        THEN
            exch$cmd_related_file_fixup (rlf_nam);

        ! Move the final related file name from the expanded string to the result string area of the related nam blo
        !
        rlf_nam [nam$b_rsl] = .rlf_nam [nam$b_esl];      !  Result name length
        rlf_nam [nam$l_rsa] = .rlf_nam [nam$l_esa];      !  Result name length
        $trace_print_fao ('related name after first parse "!AF"', .rlf_nam [nam$b_rsl], .rlf_nam [nam$l_rsa]);

        ! Initialize the structures needed by RMS $PARSE routine
        !
        $fab_init (
                fab = fab,                               ! Input file FAB
                dna = UPLIT BYTE (';*'),                 !  Default is version so that we can maximize
                dns = 2,                                 !  Default name size is two
                fna = .fil_buf,                          !  Set name addr
                fns = .fil_len,                          !  Set name size
                fop = <OFP>,                             !  Output file parse
                nam = .nam);                             !  Name block

        $nam_init (
                nam = .nam,                              ! File name block
                esa = .nam+nam$c_bln,                    !  Expanded name buffer addr
                ess = nam$c_maxrss,                      !  Expanded name size
                rlf = rlf_nam,                           !  Related name
                nop = synchk);                           !  Syntax check only

        ! Now let RMS do the dirty work for us
        !
        $trace_print_lit ('fab before related parse    <<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<<');
        $check_call (4, exch$dbg_fab_dump, fab);         ! Dump the fab
        status = $parse (FAB = fab);                     ! Parse the file spec
        $trace_print_lit ('fab after related parse    >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>');
        $check_call (4, exch$dbg_fab_dump, fab);         ! Dump the fab
        $check_call (5, exch$dbg_fab_name_print, fab);   ! Print name block information
```

```
;  1896          1977  2 ! If we don't have an expanded name we exit with the error
;  1897          1978  2 !
;  1898          1979  2 IF .nam [nam$b_esl] EQL 0
;  1899          1980  2 THEN
;  1900          1981  2     RETURN .status;
;  1901          1982  2
;  1902          1983  2 $trace_print_fao ('final result name "!AF"', .nam [nam$b_esl], .nam [nam$l_esa]);
;  1903          1984  2
;  1904          1985  2 RETURN true;
;  1905          1986  1 END;


                                                        .PSECT  EXCH$CMD_PLIT,NOWRT,2

                                        2A  3B  001E4 P.ABQ:    .ASCII  \;*\                                    ;


                                                        .PSECT  EXCH$CMD_CODE,NOWRT,2

                                              01FC 00000        .ENTRY  EXCH$CMD_RELATED_FILE_PARSE, Save R2,R3,R4,-; 1863
                                                                        R5,R6,R7,R8
                                    58 00000000G  00  9E 00002  MOVAB   SYS$PARSE, R8                           ;
                                    5E      FE50  CE  9E 00009  MOVAB   -432(SP), SP                            ;
    0050  8F          00            6E          00  2C 0000E  MOVC5   #0, (SP), #0, #80, $RMS_PTR                ; 1920
                                                B0  AD 00015                                                    ;
                              B0  AD      5003  8F  B0 00017  MOVW    #20483, $RMS_PTR                        ;
                              C6  AD            02  90 0001D  MOVB    #2, $RMS_PTR+22                         ;
                              CF  AD            02  90 00021  MOVB    #2, $RMS_PTR+31                         ;
                              D8  AD      FF50  CD  9E 00025  MOVAB   RLF_NAM, $RMS_PTR+40                    ;
                              DC  AD        10  AC  D0 0002B  MOVL    RLF_BUF, $RMS_PTR+44                    ;
                              E4  AD        0C  AC  90 00030  MOVB    RLF_LEN, $RMS_PTR+52                    ;
    0060  8F          00            6E          00  2C 00035  MOVC5   #0, (SP), #0, #96, $RMS_PTR              ; 1925
                                    FF50  CD      0003C                                                     ;
                        FF50  CD    6002  8F  B0 0003F  MOVW    #24578, $RMS_PTR                            ;
                        FF58  CD          08  90 00046  MOVB    #8, $RMS_PTR+8                              ;
                        FF5A  CD          01  8E 0004B  MNEGB   #1, $RMS_PTR+10                             ;
                        FF5C  CD          6E  9E 00050  MOVAB   RBUF, $RMS_PTR+12                           ;
                                        B0  AD 9F 00055  PUSHAB FAB                                         ; 1931
                                    68      01  FB 00058  CALLS   #1, SYS$PARSE                              ;
                                    57      50  D0 0005B  MOVL    R0, STATUS                                  ;
                                    FF5B  CD  95 0005E  TSTB    RLF_NAM+11                                  ; 1934
                                            0A  12 00062  BNEQ    1$                                          ;
                                            57  DD 00064  PUSHL   STATUS                                      ; 1936
                              00000000G     00  01  FB 00066  CALLS   #1, LIB$STOP                           ;
                                                04  0006D  RET                                               ;
                        09      86  AD          02  E1 0006E  1$:  BBC     #2, RLF_NAM+54, 2$                ; 1940
                                    FF50  CD  9F 00073  PUSHAB  RLF_NAM                                       ; 1942
                        FF08  CF          01  FB 00077  CALLS   #1, EXCH$CMD_RELATED_FILE_FIXUP             ;
                        FF53  CD    FF5B  CD  90 0007C  2$:   MOVB    RLF_NAM+11, RLF_NAM+3                   ; 1946
                        FF54  CD    FF5C  CD  D0 00083        MOVL    RLF_NAM+12, RLF_NAM+4                   ; 1947
    0050  8F          00            6E          00  2C 0008A  MOVC5   #0, (SP), #0, #80, $RMS_PTR              ; 1959
                                                B0  AD 00091                                                    ;
                              B0  AD      5003  8F  B0 00093  MOVW    #20483, $RMS_PTR                        ;
                              B4  AD  20000000  8F  D0 00099  MOVL    #536870912, $RMS_PTR+4                  ;
                              C6  AD            02  90 000A1  MOVB    #2, $RMS_PTR+22                         ;
```

```
                     CF  AD           02  90 000A5          MOVB    #2, $RMS_PTR+31
                     56      14       AC  D0 000A9          MOVL    NAM, R6
                     D8  AD           56  D0 000AD          MOVL    R6, $RMS_PTR+40
                     DC  AD       08  AC  D0 000B1          MOVL    FIL_BUF, $RMS_PTR+44
                     E0  AD     0000' CF  9E 000B6          MOVAB   P.ABQ, $RMS_PTR+48
                     E4  AD       04  AC  90 000BC          MOVB    FIL_LEN, $RMS_PTR+52
                     E5  AD           02  90 000C1          MOVB    #2, $RMS_PTR+53
 0060  8F       00      6E           00  2C 000C5          MOVC5   #0, (SP), #0, #96, (R6)                 ; 1966
                                      66  000CC
                     66      6002     8F  B0 000CD          MOVW    #24578, (R6)
                     08  A6           08  90 000D2          MOVB    #8, 8(R6)
                     0A  A6           01  8E 000D6          MNEGB   #1, 10(R6)
                     0C  A6       60  A6  9E 000DA          MOVAB   96(R6), 12(R6)
                     10  A6     FF50  CD  9E 000DF          MOVAB   RLF_NAM, 16(R6)
                         B0           AD  9F 000E5          PUSHAB  FAB                                     ; 1972
                     68               01  FB 000E8          CALLS   #1, SYS$PARSE
                     57               50  D0 000EB          MOVL    R0, STATUS
                                  0B  A6  95 000EE          TSTB    11(R6)                                  ; 1979
                                      04  12 000F1          BNEQ    3$
                     50               57  D0 000F3          MOVL    STATUS, R0                              ; 1981
                                          04 000F6          RET
                     50               01  D0 000F7 3$:      MOVL    #1, R0                                  ; 1985
                                          04 000FA          RET                                             ; 1986

; Routine Size:  251 bytes,    Routine Base:  EXCH$CMD_CODE + 0DBD
```

```
;  1907     1987  1 GLOBAL ROUTINE exch$cmd_unwind_cli_syntax (sig : $ref_bblock, mech : $ref_bblock) =     %SBTTL 'exch$cmd_unw
;  1908     1988  2 BEGIN
;  1909     1989  2 !++
;  1910     1990  2 !
;  1911     1991  2 ! FUNCTIONAL DESCRIPTION:
;  1912     1992  2 !
;  1913     1993  2 !       This routine intercepts the signal MSG$_SYNTAX.  This is used by several routines to terminate
;  1914     1994  2 !       processing when a qualifier is not defined for the particular parameter.
;  1915     1995  2 !
;  1916     1996  2 ! INPUTS:
;  1917     1997  2 !
;  1918     1998  2 !       sig  - signal argument list
;  1919     1999  2 !       mech - mechanism argument list
;  1920     2000  2 !
;  1921     2001  2 ! IMPLICIT INPUTS:
;  1922     2002  2 !
;  1923     2003  2 !       none
;  1924     2004  2 !
;  1925     2005  2 ! OUTPUTS:
;  1926     2006  2 !
;  1927     2007  2 !       none
;  1928     2008  2 !
;  1929     2009  2 ! IMPLICIT OUTPUTS:
;  1930     2010  2 !
;  1931     2011  2 !       none
;  1932     2012  2 !
;  1933     2013  2 ! ROUTINE VALUE:
;  1934     2014  2 !
;  1935     2015  2 !       SS$_UNWIND if signal was MSG$_SYNTAX, otherwise SS$_RESIGNAL.
;  1936     2016  2 !
;  1937     2017  2 ! SIDE EFFECTS:
;  1938     2018  2 !
;  1939     2019  2 !       If we unwind, the control flow will return as if a RETURN had been made from the enabling routine
;  1940     2020  2 !--
;  1941     2021  2 $dbgtrc_prefix ('cmd_unwind_cli_syntax> ');
;  1942     2022  2
;  1943     2023  2 LOCAL
;  1944     2024  2     status
;  1945     2025  2     ;
;  1946     2026  2
;  1947     2027  2 ! If the signal name is what we are looking for, then do a default VMS unwind
;  1948     2028  2 !
;  1949     2029  2 IF .sig [chf$l_sig_name] EQL msg$_syntax        ! DCL CLI error message (sinful knowlege of what DCL does!)
;  1950     2030  2 THEN
;  1951     2031  3     BEGIN
;  1952     2032  3     $debug_print_lit ('unwinding');
;  1953     2033  3     mech [chf$l_mch_savr0] = -1;                ! Flag R0 so that we will know msg$_syntax occurred
;  1954     2034  4     IF NOT (status = $unwind ())
;  1955     2035  3     THEN
;  1956     2036  3         $exch_signal_stop (.status);
;  1957     2037  2     END;
;  1958     2038  2
;  1959     2039  2 RETURN ss$_resignal;
;  1960     2040  1 END;
```

```
                                                        .EXTRN  SYS$UNWIND

                                              0000 00000        .ENTRY  EXCH$CMD_UNWIND_CLI_SYNTAX, Save nothing    ; 1987
                          50        04    AC  D0 00002        MOVL    SIG, R0                                     ; 2029
                000310FC  8F        04    A0  D1 00006        CMPL    4(R0), #200956                              ;
                                          1E  12 0000E        BNEQ    1$                                          ;
                          50        08    AC  D0 00010        MOVL    MECH, R0                                    ; 2033
                      0C  A0              01  CE 00014        MNEGL   #1, 12(R0)                                  ;
                                          7E  7C 00018        CLRQ    -(SP)                                       ; 2034
                00000000G 00              02  FB 0001A        CALLS   #2, SYS$UNWIND                              ;
                          0A              50  E8 00021        BLBS    STATUS, 1$                                  ;
                                          50  DD 00024        PUSHL   STATUS                                      ; 2036
                00000000G 00              01  FB 00026        CALLS   #1, LIB$STOP                                ;
                                              04 0002D        RET                                                 ;
                          50        0918  8F  3C 0002E 1$:     MOVZWL  #2328, R0                                   ; 2039
                                              04 00033        RET                                                 ; 2040

; Routine Size:  52 bytes,    Routine Base:  EXCH$CMD_CODE + 0EB8
```

```
; 1962          2041  1 END
; 1963          2042  0 ELUDOM


                                                        .EXTRN  LIB$SIGNAL, LIB$STOP

;                                PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  EXCH$CMD_PLIT                  486  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  EXCH$CMD_CODE                 3820  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                             Library Statistics
;
;                                      -------- Symbols --------      Pages       Processing
;       File                           Total    Loaded   Percent      Mapped      Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1        18619       122         0        1000        00:01.9
;  _$255$DUA28:[EXCHNG.OBJ]EXCLIB.L32;1  1151       129        11          79        00:01.4


; Information:  3
; Warnings:     0
; Errors:       0


;                                    COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:EXCCMD/OBJ=OBJ$:EXCCMD MSRC$:EXCCMD/UPDATE=(ENH$:EXCCMD)

; Size:          3820 code + 486 data bytes
; Run Time:         01:14.1
; Elapsed Time:     03:39.7
; Lines/CPU Min:     1654
; Lexemes/CPU-Min: 28223
; Memory Used:  336 pages
; Compilation Complete
```

EXCREQ
R32

MAILCVT
COM

SYSGTTSTR
MSG

EXCOPY
LIS

USSLNK
COM

EXCDEFS
SDL

EXCLIB
B32

EXCCLDTBL
LIS

MAILUAF
COM

USSTSTLNK
COM

EXCHNG

XATEST
COM

EXCHANGE
MAP

LABIO
OPT

MSCPMOUNT
COM

LABIOCIN
OPT

EXCCMD
CLS

DRCOPY
PRM